

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Attorney Docket No. 067238/0115

#3
Priority
Paper
MHA
5/13/02

Applicant: Atsufumi SHIBAYAMA, et al.
Title: DATA PROCESSOR WITH AN IMPROVED DATA DEPENDENCE
DETECTOR
Serial No.: 09/994,806
Filed: November 28, 2001
Examiner: Unassigned
Art Unit: 2152

CLAIM FOR CONVENTION PRIORITY

Commissioner for Patents
Washington, D.C. 20231

Sir:

The benefit of the filing date of the following prior foreign application filed in the following foreign country is hereby requested, and the right of priority provided in 35 U.S.C. § 119 is hereby claimed.

In support of this claim, filed herewith is a certified copy of said original foreign application:

Japan Patent Application No. 2000-363727 filed 11/29/2000.

Respectfully submitted,



Ankur D. Shah
Registration No. 41,514

JANUARY 18, 2002

Date

FOLEY & LARDNER
Washington Harbour
3000 K Street, N.W. – Suite 500
Washington, D.C. 20007
Telephone: (202) 672-5300
Facsimile: (202) 672-5399

日本国特許庁
JAPAN PATENT OFFICE

US

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日

Date of Application:

2000年11月29日

出願番号

Application Number:

特願2000-363727

出願人

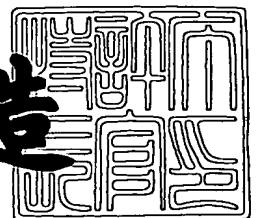
Applicant(s):

日本電気株式会社

2001年 8月31日

特許庁長官
Commissioner,
Japan Patent Office

及川耕造



出証番号 出証特2001-3077813

【書類名】 特許願

【整理番号】 34002109

【提出日】 平成12年11月29日

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 9/46
G06F 15/16

【発明者】

【住所又は居所】 東京都港区芝五丁目7番1号 日本電気株式会社内

【氏名】 柴山 充文

【発明者】

【住所又は居所】 東京都港区芝五丁目7番1号 日本電気株式会社内

【氏名】 松下 智

【発明者】

【住所又は居所】 東京都港区芝五丁目7番1号 日本電気株式会社内

【氏名】 鳥居 淳

【発明者】

【住所又は居所】 東京都港区芝五丁目7番1号 日本電気株式会社内

【氏名】 西 直樹

【特許出願人】

【識別番号】 000004237

【氏名又は名称】 日本電気株式会社

【代理人】

【識別番号】 100080816

【弁理士】

【氏名又は名称】 加藤 朝道

【電話番号】 045-476-1131

【手数料の表示】

【予納台帳番号】 030362

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9304371

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 データ依存関係検出装置

【特許請求の範囲】

【請求項 1】

非プログラム順序でメモリ操作命令を実行するに際して、前記メモリ操作命令の処理アドレス間の依存関係の有無を検出するデータ依存関係検出装置において

前記メモリ操作命令の処理アドレス間の依存関係が存在しない場合にも依存関係が存在する旨を検出する場合があることを許容する構成とされている、ことを特徴とする、データ依存関係検出装置。

【請求項 2】

非プログラム順序でメモリ操作命令を実行するに際して、前記メモリ操作命令の処理アドレス間の依存関係の有無を検出するデータ依存関係検出装置において

前記メモリ操作命令の処理アドレス間の依存関係が存在する場合には必ず依存関係が存在する旨を検出すると共に、前記依存関係が存在しない場合にも依存関係が存在する旨を検出する場合があることを許容する構成とされている、ことを特徴とする、データ依存関係検出装置。

【請求項 3】

非プログラム順序でメモリ操作命令を実行するに際して、前記メモリ操作命令の処理アドレス間の依存関係の有無を検出するデータ依存関係検出装置において

前記メモリ操作命令の処理アドレスを受け該処理アドレスを一意的番号に変換するアドレス変換手段と、

前記アドレス変換手段で変換された番号に対応して、メモリ操作命令が実行されたか否かを示す値を記憶する複数の記憶手段を有する実行履歴記憶手段と、

を少なくとも含み、

前記メモリ操作命令の処理アドレス間の依存関係が存在する場合には必ず依存関係が存在する旨を検出すると共に、前記依存関係が存在しない場合にも依存関

係が存在する旨を検出する場合がある、ことを許容する、ことを特徴とする、データ依存関係検出装置。

【請求項4】

請求項3記載のデータ依存関係検出装置において、

前記メモリ操作命令のロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、該ロード命令が実行されたことを示す値を格納し、

前記メモリ操作命令のストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているロード命令が実行されたか否かを示す値を読み出すことで、前記実行されたストア命令から、前記実行されたロード命令への正依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項5】

請求項3記載のデータ依存関係検出装置において、

前記メモリ操作命令のストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、該ストア命令が実行されたことを示す値を格納し、

前記メモリ操作命令のロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているストア命令が実行されたか否かを示す値を読み出すことで、前記実行されたロード命令から、前記実行されたストア命令への逆依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項6】

請求項 3 記載のデータ依存関係検出装置において、

前記メモリ操作命令のストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、ストア命令が実行されたか否かを示す値を読み出すことに加えて、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、該ストア命令が実行されたことを示す値を格納することで、前記実行されたストア命令間に出力依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 7】

請求項 3 記載のデータ依存関係検出装置において、

前記メモリ操作命令のロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、前記メモリ操作命令のストア命令が実行されたか否かを示す値を読み出すことに加えて、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段にロード命令が実行されたことを示す値を格納し、

ストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているロード命令が実行されたか否かを示す値及びストア命令が実行されたか否かを示す値を読み出すことに加えて、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段にストア命令が実行されたことを示す値を格納し、前記実行されたストア命令から、前記実行されたロード命令への正依存関係が存在する可能性、及び前記実行されたロード命令から、前記実行されたストア命令への逆依存関係が存在する可能性、及び前記実行されたストア命令間の出力依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 8】

前記アドレス変換手段は、入力されるアドレスの所定のビットをそのまま出力することで、任意のアドレスを前記所定のビットで規定される番号に変換する、構成とされている、ことを特徴とする、請求項 3 から請求項 7 のいずれかに記載のデータ依存関係検出装置。

【請求項 9】

前記アドレス変換手段は、入力されるアドレスの複数の所定のビットの排他的論理和を出力する手段を備え、任意のアドレスを前記排他的論理和出力で規定される番号に変換する、構成とされている、ことを特徴とする、請求項 3 から請求項 7 のいずれかに記載のデータ依存関係検出装置。

【請求項 1 0】

複数のプロセッサから構成されるマルチプロセッサシステムにおける前記各々のプロセッサが備えるデータ依存関係検出装置であって、前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスと、自プロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスとの間の依存関係の有無を検出するデータ依存関係検出装置において、

依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する構成とされている、ことを特徴とする、データ依存関係検出装置。

【請求項 1 1】

複数のプロセッサから構成されるマルチプロセッサシステムにおける前記各々のプロセッサが備えるデータ依存関係検出装置であって、前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスと、自プロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスとの間の依存関係の有無を検出するデータ依存関係検出装置において、

前記依存関係が存在する場合は必ず依存関係が存在する旨を検出すると共に、前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する構成とされている、ことを特徴とする、データ依存関係検出装置

【請求項 1 2】

複数のプロセッサから構成されるマルチプロセッサシステムにおける前記各々のプロセッサが備えるデータ依存関係検出装置であって、前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスと、自プロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスとの間の依存関係の有無を検出するデータ依存関係検出装置において、

自プロセッサ及び他プロセッサが実行するメモリ操作命令の処理アドレスを一意的な番号に変換する複数のアドレス変換手段と、

前記変換された番号に対応して、メモリ操作命令が実行されたか否かを記憶する複数の記憶手段から構成される実行履歴記憶手段と、

を少なくとも含み、前記依存関係が存在する場合は必ず依存関係が存在する旨を検出すると共に、前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する、ことを特徴とする、データ依存関係検出装置。

【請求項 1 3】

請求項 1 2 記載のデータ依存関係検出装置において、

自プロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意的な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでロード命令が実行されたことを示す値を格納し、他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意的な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでロード命令が実行されたか否かを示す値を読み出すことで、前記他のプロセッサで実行されたストア命令から、前記自プロセッサで実行されたロード命令への正依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 14】

請求項 12 記載のデータ依存関係検出装置において、

自プロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでストア命令が実行されたことを示す値を格納し、他のプロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでストア命令が実行されたか否かを示す値を読み出すことで、前記他のプロセッサで実行されたロード命令から、前記自プロセッサで実行されたストア命令への逆依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 15】

請求項 12 記載のデータ依存関係検出装置において、

他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、自プロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでストア命令が実行されたことを示す値を格納することで、前記他のプロセッサで実行されたストア命令から、前記自プロセッサで実行されたストア命令への出力依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 16】

請求項 12 記載のデータ依存関係検出装置において、

他のプロセッサでロード命令が実行された場合には、該ロード命令が処理対象

とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでロード命令が実行されたか否かを示す値及び自プロセッサでストア命令が実行されたか否かを示す値を読み出し、自プロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでロード命令が実行されたことを示す値を格納し、自プロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでストア命令が実行されたことを示す値を格納することで、前記他プロセッサで実行されたストア命令から、前記自プロセッサで実行されたロード命令への正依存関係が存在する可能性、及び前記他プロセッサで実行されたロード命令から、前記自プロセッサで実行されたストア命令への逆依存関係が存在する可能性、及び前記他のプロセッサで実行されたストア命令から、前記自プロセッサで実行されたストア命令への出力依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 17】

前記アドレス変換手段が、入力されるアドレスの所定のビットをそのまま出力することで、任意のアドレスを前記所定のビットで規定される番号に変換する、構成とされている、ことを特徴とする、請求項 12 から請求項 16 のいずれかに記載のデータ依存関係検出装置。

【請求項 18】

前記アドレス変換手段が、入力されるアドレスの複数の所定のビットの排他的

論理和を出力する手段を備え、任意のアドレスを前記排他的論理和出力で規定される番号に変換する、構成とされている、ことを特徴とする、請求項12から請求項16のいずれか一に記載のデータ依存関係検出装置。

【請求項19】

請求項13から請求項16のいずれか一に記載のデータ依存関係検出装置において、

他のプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前に位置するスレッドを実行するプロセッサにおいてメモリ操作命令が実行された場合にのみ、該メモリ操作命令の処理アドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているメモリ操作命令が実行されたか否かを示す値を読み出す、ことを特徴とする、データ依存関係検出装置。

【請求項20】

請求項13から請求項16のいずれか一に記載のデータ依存関係検出装置において、

他のプロセッサがメモリ操作命令を実行する場合に、該メモリ操作命令の処理アドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているメモリ操作命令が実行されたか否かを示す値を読み出す際に、自プロセッサが実行するスレッドよりもプログラム順序で前に位置するスレッドを実行するプロセッサが実行したメモリ操作命令により読み出された値のみを参照する、ことを特徴とする、データ依存関係検出装置。

【請求項21】

複数のプロセッサから構成されるマルチプロセッサシステムにおいて、前記各々のプロセッサが備えるデータ依存関係検出装置であって、かつ前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスと、自プロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスとの間の依存関係の有無、及び自プ

ロセッサが実行するスレッドが含むメモリ操作命令の対象アドレス間の依存関係を検出するデータ依存関係検出装置において、

前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する構成とされている、ことを特徴とする、データ依存関係検出装置。

【請求項 2 2】

複数のプロセッサから構成されるマルチプロセッサシステムにおいて、前記各々のプロセッサが備えるデータ依存関係検出装置であって、かつ前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスと、自プロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスとの間の依存関係の有無、及び自プロセッサが実行するスレッドが含むメモリ操作命令の対象アドレス間の依存関係を検出するデータ依存関係検出装置において、

前記依存関係が存在する場合は必ず依存関係が存在する旨を検出すると共に、前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する、構成とされている、ことを特徴とする、データ依存関係検出装置。

【請求項 2 3】

複数のプロセッサから構成されるマルチプロセッサシステムにおいて、前記各々のプロセッサが備えるデータ依存関係検出装置であって、かつ前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスと、自プロセッサが実行するスレッドが含むメモリ操作命令の処理アドレスとの間の依存関係の有無、及び自プロセッサが実行するスレッドが含むメモリ操作命令の対象アドレス間の依存関係を検出するデータ依存関係検出装置において、

自プロセッサ及び他プロセッサが実行するメモリ操作命令の処理アドレスを一意な番号に変換する複数のアドレス変換手段と、

前記変換された番号に対応して、メモリ操作命令が実行されたか否かを記憶する複数の記憶手段から構成される実行履歴記憶手段と、を少なくとも含み、

前記依存関係が存在する場合は必ず依存関係が存在する旨を検出すると共に、前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する、ことを特徴とする、データ依存関係検出装置。

【請求項 2 4】

請求項 2 3 記載のデータ依存関係検出装置において、

自プロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでロード命令が実行されたことを示す値を格納し、

自プロセッサあるいは他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでロード命令が実行されたか否かを示す値を読み出すことで、前記自プロセッサあるいは他のプロセッサで実行されたストア命令から、前記自プロセッサで実行されたロード命令への正依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 2 5】

請求項 2 3 記載のデータ依存関係検出装置において、

自プロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでストア命令が実行されたことを示す値を格納し、

自プロセッサあるいは他のプロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでストア命令が実行されたか否かを示す値を読み出すことで、前記自プロセッサあるいは他のプロセッサで実行されたロード命令から、前記自プロセッサで実行されたストア命令

への逆依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 2 6】

請求項 2 3 記載のデータ依存関係検出装置において、

他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている自プロセッサでストア命令を実行したか否かを示す値を読み出し、

自プロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでストア命令を実行したか否かを示す値を読み出すのに加えて、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に、自プロセッサでストア命令が実行されたことを示す値を格納することで、前記自プロセッサあるいは他のプロセッサで実行されたストア命令から、前記自プロセッサで実行されたストア命令への出力依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 2 7】

請求項 2 3 記載のデータ依存関係検出装置において、

自プロセッサあるいは他のプロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されている、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、

自プロセッサあるいは他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換さ

れた番号に対応する記憶手段に格納されている、自プロセッサでロード命令が実行されたか否かを示す値及び自プロセッサでストア命令が実行されたか否かを示す値を読み出すことに加えて、自プロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に自プロセッサでロード命令が実行されたことを示す値を格納し、

自プロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に自プロセッサでストア命令が実行されたことを示す値を格納することで、前記自プロセッサあるいは他プロセッサで実行されたストア命令から、前記自プロセッサで実行されたロード命令への正依存関係が存在する可能性、及び、前記自プロセッサあるいは他プロセッサで実行されたロード命令から、前記自プロセッサで実行されたストア命令への逆依存関係が存在する可能性、及び前記自プロセッサあるいは他のプロセッサで実行されたストア命令から、前記自プロセッサで実行されたストア命令への出力依存関係が存在する可能性を検出する、ことを特徴とする、データ依存関係検出装置。

【請求項 2 8】

前記アドレス変換手段が、入力されるアドレスの所定のビットをそのまま出力することで、任意のアドレスを前記所定のビットで規定される番号に変換する、構成とされている、ことを特徴とする、請求項 2 3 から請求項 2 7 のいずれかに記載のデータ依存関係検出装置。

【請求項 2 9】

前記アドレス変換手段が、入力されるアドレスの複数の所定位置のビットの排他的論理和を出力する手段を備え、任意のアドレスを前記排他的論理和出力で規定される番号に変換する、構成とされている、ことを特徴とする、請求項 2 3 から請求項 2 7 のいずれかに記載のデータ依存関係検出装置。

【請求項 3 0】

請求項 2 4 から請求項 2 7 のいずれか一に記載のデータ依存関係検出装置において、

他のプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前に位置するスレッドを実行するプロセッサにおいてメモリ操作命令が実行された場合にのみ、該メモリ操作命令の処理アドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているメモリ操作命令が実行されたか否かを示す値を読み出す、ことを特徴とする、データ依存関係検出装置。

【請求項 3 1】

請求項 2 4 から請求項 2 7 のいずれか一に記載のデータ依存関係検出装置において、

他のプロセッサがロード命令またはストア命令を実行する場合に、該ロード命令または該ストア命令が処理対象とするアドレスを前記アドレス変換手段により一意な番号に変換し、前記実行履歴記憶手段を構成する複数の記憶手段のうち、前記変換された番号に対応する記憶手段に格納されているロード命令またはストア命令が実行されたか否かを示す値を読み出す際に、自プロセッサが実行するスレッドよりもプログラム順序で前に位置するスレッドを実行するプロセッサが実行したロード命令またはストア命令により読み出された値のみを参照する、ことを特徴とする、データ依存関係検出装置。

【請求項 3 2】

請求項 1 から請求項 3 1 のいずれか一に記載のデータ依存関係検出装置を搭載した半導体集積回路装置。

【請求項 3 3】

前記メモリ操作命令の処理アドレス間の依存関係が存在しない場合にも依存関係が存在する旨を検出する場合があることを許容する構成が、複数のエントリよりなり、前記メモリ操作命令の命令履歴を管理する実行履歴記憶手段へのエントリ番号を前記メモリ操作命令のアドレスから生成するアドレス変換手段を含み、

前記アドレス変換回路は、前記メモリ操作命令の処理対象アドレスを、前記実

行履歴記憶手段の総エントリ数の範囲内のエントリ番号に変換し、同一の対象アドレスに対しては同一のエントリ番号を出力するとともに、同一のエントリ番号に対して異なる対象アドレスが重複して存在することが許容される構成とされ、前記メモリ操作命令の処理アドレス間の依存関係が存在しない場合にも、依存関係が存在する旨を検出するものである、ことを特徴とする、請求項1、2、11、12、21、22のいずれかに記載のデータ依存関係検出装置。

【請求項34】

前記アドレス変換手段が、複数のエントリよりなり、前記メモリ操作命令の命令履歴を管理する実行履歴記憶手段へのエントリ番号を、前記メモリ操作命令のアドレスから生成するにあたり、前記メモリ操作命令の処理対象アドレスを、前記実行履歴記憶手段の総エントリ数の範囲内のエントリ番号に変換し、同一の対象アドレスに対しては同一のエントリ番号を出力するとともに、同一のエントリ番号に対して異なる対象アドレスが重複して存在することが許容される構成とされている、ことを特徴とする、請求項3乃至9、請求項12乃至20、請求項23乃至31のいずれかに記載のデータ依存関係検出装置。

【請求項35】

命令実行部と、プロセッサ制御部と、データ依存関係検出装置とを少なくとも含むプロセッサ装置において、

前記プロセッサ制御部は、前記プロセッサ装置が命令を実行するにあたり、実行する命令の種別を前記命令実行部、及び、前記データ依存関係検出装置に出力し、その際、実行する命令が、メモリから汎用レジスタへのデータの転送を行うロード命令、または、汎用レジスタからメモリへのデータ転送を行うストア命令よりなるメモリ操作命令である場合には、該メモリ操作命令の処理対象アドレスを、ロード・ストアアドレス線から、前記命令実行部、及び、前記データ依存関係検出装置に出力するとともに、ロード命令またはストア命令をデータ依存投機実行する場合に、その旨を、投機実行フラグを通じて、前記データ依存関係検出装置に通知する構成とされ、

前記データ依存関係検出装置は、前記プロセッサ制御部から出力されるロード命令とストア命令の処理対象アドレスを、前記ロード・ストアアドレス線から入

力し、前記対象アドレスを、前記対象アドレスのビット幅よりも小のビット幅で表現されるエントリ番号に変換し、ロード命令とストア命令について同一の対象アドレスに対しては同一のエントリ番号を出力し、異なる対象アドレスに対して同一のエントリ番号が出力され、同一のエントリ番号に対して異なる対象アドレスが存在する場合がある構成よりなるハッシュ関数回路と、

複数のエントリから構成され、前記ハッシュ関数回路が出力するエントリ番号が指し示すエントリに対して、前記プロセッサ制御部が出力する前記実行命令と前記投機実行フラグとを参照して、該エントリの書き込みや読み出し処理を行う命令履歴テーブルと、

を備え、データ依存投機実行が失敗したか成功したかを示す値をデータ依存検出結果に出力して前記プロセッサ制御部に通知する、ことを特徴とする、プロセッサ装置。

【請求項 3 6】

前記プロセッサ装置がデータ依存関係に対して投機的な命令実行を行う状態にある場合、投機的なロード命令が実行されると、前記データ依存関係検出装置は、投機的なロード命令の対象アドレスを、前記ハッシュ関数回路によりエントリ番号に変換して前記命令履歴テーブルに入力し、

前記命令履歴テーブルは、入力されたエントリ番号に対応するエントリを、投機的なロード命令が実行されたことを示す値に設定し、

投機的なストア命令が実行された場合には、ストア命令の対象アドレスを前記ハッシュ関数回路によりエントリ番号に変換して前記命令履歴テーブルに入力し、

前記命令履歴テーブルは、入力されたエントリ番号に対応するエントリに格納されている値を読み出し、前記読み出した値に基づいて、ストア命令から実行されたロード命令へ正依存関係が存在する可能性があるか否かを検出し、正依存関係が存在する可能性があることが検出された場合、前記データ依存関係検出装置は、データ依存投機実行が失敗したことを示す値をデータ依存検出結果に出力する、ことを特徴とする、請求項 3 5 に記載のプロセッサ装置。

【請求項 3 7】

前記データ依存関係検出装置は、前記プロセッサ制御部から前記データ依存関係検出装置に入力される前記投機実行フラグに基づいて、前記命令履歴テーブルのすべてのエントリに同一の値を書き込む初期化機能を有する、ことを特徴とする、請求項 3 5 又は 3 6 に記載のプロセッサ装置。

【請求項 3 8】

前記データ依存関係検出装置からデータ依存投機実行が失敗したことを示す値が前記データ依存検出結果に出力された場合、前記プロセッサ制御部及び前記命令実行部は、データ依存投機実行が失敗したことに對する回復処理を実行する、ことを特徴とする、請求項 3 5 又は 3 6 に記載のプロセッサ装置。

【請求項 3 9】

前記データ依存関係検出装置は、ストア命令からロード命令へ、正依存関係が存在しないことが検出された場合、データ依存投機実行が成功したことを示す値を前記データ依存検出結果に出力し、これを受けて前記プロセッサ制御部はデータ依存投機実行が成功したとして、データ依存投機実行が失敗したことに對する回復処理は実行せずに、後続の命令実行を続行する、ことを特徴とする、請求項 3 5 に記載のプロセッサ装置。

【請求項 4 0】

プロセッサ制御部、データ依存関係検出装置、及び命令実行部を含むプロセッサを複数（N 個）備え、

プログラムをプロセッサによってスレッド並列処理するにあたり、前記複数のプロセッサの各々が実行すべきスレッドの割り当てを行い、前記複数のプロセッサのそれぞれに、スレッド実行がデータ依存投機実行か否かを示す投機実行フラグを出力し、さらに一のプロセッサと他のプロセッサが実行するスレッド間の順序関係を示すスレッド順序を出力するスレッド制御部と、

前記各プロセッサは、それぞれ、投機実行フラグ及びスレッド順序を入力し、スレッド制御部より割り当てられたスレッドを、各プロセッサが実行する場合、各プロセッサの前記プロセッサ制御部は、自プロセッサが実行する命令の種別を、実行命令線から自プロセッサの前記命令実行部と前記データ依存関係検出装置とに出力するとともに、実行命令線を通じて他のプロセッサに出力し、

前記データ依存関係検出装置は、自プロセッサが実行する命令種別を実行命令線から入力するとともに、他のプロセッサが実行する命令種別を、前記他のプロセッサの前記プロセッサ制御部から出力される実行命令線を通じてそれぞれ入力し、

実行する命令がロード命令またはストア命令の場合、前記プロセッサ制御部は、ロード命令またはストア命令の対象アドレスを、ロード・ストアアドレス線から、自プロセッサの前記命令実行部と前記データ依存関係検出装置に出力するとともに、ロード・ストアアドレス線を通じて他プロセッサに出力し、

前記データ依存関係検出装置は、プロセッサ個数分（N個）のハッシュ関数回路と、命令履歴テーブルと、論理和回路とを備え、

自プロセッサの前記プロセッサ制御部からの入力として、実行命令線、ロード・ストアアドレス線を入力するとともに、及び、他のプロセッサの前記プロセッサ制御部から、それぞれの実行命令線とロード・ストアアドレス線とを入力とし

前記スレッド制御部からは投機実行フラグを入力し、前記投機実行フラグに基づき、自プロセッサが確定実行状態または投機実行状態のどちらにあるかを判断し、

自プロセッサが実行するスレッドが、他のプロセッサが実行するそれぞれのスレッドよりもプログラム順序で前の順序にあるか否かは、前記スレッド制御部から前記複数のプロセッサの前記データ依存関係検出装置にそれぞれ入力されるスレッド順序で判断し、さらに、前記スレッド制御部へは、データ依存検出結果を出力し、

前記ハッシュ関数回路は、ロード命令またはストア命令の対象アドレスを、命令履歴テーブルの総エントリ数の範囲内のエントリ番号に変換するハッシュ関数を実現する論理回路からなり、同一の入力に対しては同一の値をもち、前記ハッシュ関数回路のうちの一つは、自プロセッサが実行するロード命令の対象アドレスを命令履歴テーブルのエントリ番号に変換し、残りのN-1個の前記ハッシュ関数回路はそれぞれ、対応する他のプロセッサが実行するストア命令の対象アドレスを対応する命令履歴テーブルのエントリ番号に変換し、

前記命令履歴テーブルは、複数のエントリから構成され、一の書き込みポートと $N - 1$ 個の読み出しポートとを備え、

自プロセッサが実行するロード命令の対象アドレスを入力とする前記ハッシュ関数回路の出力は、前記命令履歴テーブルの書き込みポートに接続され、前記ハッシュ関数回路が出力するエントリ番号が指し示すエントリに対して書き込み処理を行い、

他のプロセッサが実行するストア命令の対象アドレスを入力とする前記ハッシュ関数回路の出力は、対応する命令履歴テーブルの読み出しポートに接続され、前記ハッシュ関数回路が出力するエントリ番号が指し示すエントリに対して読み出し処理を行い、

前記論理和回路は、前記命令履歴テーブルが備える $N - 1$ 個の読み出しポートの $N - 1$ 個の読み出し結果に対して論理和を演算した結果をデータ依存検出結果として前記スレッド制御部に出力する、ことを特徴とする、マルチプロセッサ装置。

【請求項 4 1】

前記スレッド制御部から前記データ依存関係検出装置に入力される前記投機実行フラグに基づいて前記命令履歴テーブルのすべてのエントリに同一の値を書き込む初期化機能を有する、ことを特徴とする、請求項 4 0 記載のマルチプロセッサ装置。

【請求項 4 2】

自プロセッサが投機実行状態にあるときに、投機的なロード命令が実行されると、前記データ依存関係検出装置は、ロード命令の対象アドレスを、対応する一の前記ハッシュ関数回路により前記命令履歴テーブルのエントリ番号に変換して前記命令履歴テーブルの書き込みポートに入力し、前記命令履歴テーブルは、入力されたエントリ番号に対応するエントリを、投機的なロード命令が実行されたことを示す値に設定し、

自プロセッサが投機実行状態にあるときに、他のプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前の順序にあるスレッド、すなわち先行するスレッドを割り当てられたプロセッサでストア命令が実行された場

合には、ストア命令の対象アドレスを前記ハッシュ関数回路により前記命令履歴テーブルのエントリ番号に変換し、前記命令履歴テーブルの $N-1$ 個の入力ポートのうち対応するものに入力し、前記命令履歴テーブルは、入力されたエントリ番号に対応するエントリの内容を読み出して前記論理和回路に出力し、

前記論理和回路は、前記命令履歴テーブルの $N-1$ の読み出しポートの出力の論理和を演算し前記データ依存検出結果として前記スレッド制御部に出力する、ことを特徴とする、請求項40又は41に記載のマルチプロセッサ装置。

【請求項43】

前記複数（ N 個）のプロセッサにおいて、一のプロセッサは、他のプロセッサのうち一のプロセッサが実行するスレッドよりも先行するスレッドを実行するプロセッサのいずれかが実行したストア命令の対象アドレスが、前記一のプロセッサがそれまでデータ依存投機実行したロード命令の対象アドレスと等しいか、あるいは等しくない場合でも、前記命令履歴テーブルの同じエントリに割り当てられた場合、前記命令履歴テーブルからデータ依存投機実行されたロード命令が存在することを示す値が読み出され、正依存関係が存在する可能性があることが検出され、前記一のプロセッサの前記データ依存関係検出装置は、データ依存投機実行が失敗したことを示す値を、前記データ依存検出結果を通じて、前記スレッド制御部に出力し、

前記スレッド制御部は、 N 個のプロセッサのいずれかより、データ依存投機実行の失敗の通知を受け取ると、その失敗通知を出力したプロセッサ、及び、その失敗通知を出力したプロセッサが実行していたスレッドよりも、プログラム順序で後に位置するスレッドを実行しているプロセッサに対して、データ依存投機実行の失敗による回復処理の実行要求を出力し、

データ依存投機実行の失敗による回復処理の実行要求の出力対象となった各プロセッサでは、データ依存投機実行の失敗による回復処理の実行要求が、回復処理実行要求信号を通じて、前記プロセッサ制御部に通知される、ことを特徴とする、請求項40又は41に記載のマルチプロセッサ装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、マイクロプロセッサ等のデータ処理装置に関し、特にマイクロプロセッサの性能向上技術に関する。

【0002】

【従来の技術】

従来より、マイクロプロセッサの性能向上技術として、マイクロプロセッサが実行するプログラム中の命令の順序とは異なる順序で命令の実行を行う非プログラム順序実行（アウトオブオーダー実行）技術が広く使われている。プロセッサが非プログラム順序実行を行う場合、実行可能になった命令から実行を行うため、プログラム中の命令の順序で実行を行うプログラム順序実行（インオーダー実行）を行うのに比べて、プロセッサの性能を向上させることが可能となる。

【0003】

しかしながら、非プログラム順序実行は無条件に行えるわけではない。非プログラム順序実行が可能な条件として、まず、実行する命令間のレジスタに関する正依存関係がないことがあげられる。

【0004】

プログラム順序で前にある命令が変更を行うレジスタを、プログラム順序でそれより後にある命令が参照する場合、前にある命令から後にある命令へレジスタに関する正依存関係があるという。この場合、それら2つの命令をプログラム順序とは逆の順序で実行を行うと、プログラムの意味が変わってしまい、正しいプログラムの実行結果を得ることができない。すなわち、レジスタに関して正依存関係がある場合、非プログラム順序で実行を行うことができない。

【0005】

同様に、プログラム順序で前にある命令が参照するレジスタを、プログラム順序でそれより後にある命令が変更を行う場合、前にある命令から後にある命令へレジスタに関する逆依存関係があるという。

【0006】

また、プログラム順序で前にある命令が変更を行うレジスタを、プログラム順序でそれより後にある命令も変更を行う場合、前にある命令から後にある命令へ

レジスタに関する出力依存関係があるという。

【0007】

いずれの場合も、それら2つの命令をプログラム順序とは逆の順序で実行を行うと、プログラムの意味が変わってしまい、正しいプログラムの実行結果を得ることができない。

【0008】

通常、各命令が変更あるいは参照の対象とするレジスタは、命令のデコード時に判明するため、正依存関係のある命令に対しては、非プログラム順序で実行を行わないことで対応することが多い。

【0009】

また、逆依存関係、出力依存関係に対しては、レジスタリネーミング等の技術で依存を解消して、非プログラム順序実行を可能にすることも行われている。

【0010】

メモリに対して操作を行う命令（「メモリ操作命令」という）に対しては、非プログラム順序実行に際して、レジスタに関する依存関係に加えて、メモリに関する依存関係に対して考慮が必要となる。

【0011】

通常、メモリ操作命令には、メモリからデータを読み出すロード命令と、メモリへデータを書き込むストア命令が含まれる。

【0012】

プログラム中に出現する、ある2つのロード命令またはストア命令が対象とするアドレスが互いに異なれば、メモリの異なる位置が読み出しまたは書き込みの対象となるため、それら2つのロード命令またはストア命令の間にはメモリに関して依存関係は存在しない。すなわち、それら2つの命令は非プログラム順序で実行することが可能である。

【0013】

一方、プログラム中に出現するある2つのロード命令またはストア命令が対象とするアドレスが同じ場合、メモリの同じ位置が読み出しまたは書き込みの対象となるため、それら2つのロード命令またはストア命令間には、メモリに関して

依存関係が存在する。例えば、プログラム順序で前にあるストア命令が書き込みを行うアドレスに対して、プログラム順序でそれより後にあるロード命令が読み出しを行う場合、前にあるストア命令から後にあるロード命令へメモリに関する正依存関係があるという。この場合、それら2つのロード命令、ストア命令をプログラム順序とは逆の順序で実行を行うと、プログラムの意味が変わってしまい、正しいプログラムの実行結果を得ることができない。すなわち、メモリに関して正依存関係がある場合、非プログラム順序で実行を行うことができない。

【0014】

同様にして、プログラム順序で前にあるロード命令が読み出しを行うアドレスに対して、プログラム順序でそれより後にあるストア命令が書き込みを行う場合、前にあるロード命令から後にあるストア命令へメモリに関する逆依存関係があるという。

【0015】

また、プログラム順序で前にあるストア命令が書き込みを行うアドレスに対して、プログラム順序でそれより後にあるストア命令も書き込みを行う場合、前にあるストア命令から後にあるストア命令へメモリに関する出力依存関係があるという。

【0016】

いずれの場合も、それら2つのロード・ストア命令をプログラム順序とは逆の順序で実行を行うと、プログラムの意味が変わってしまい、正しいプログラムの実行結果を得ることができない。

【0017】

メモリに関する逆依存関係及び出力依存関係に対しては、ストア命令が書き込みを行うデータを一時的なバッファ（このバッファは、通常、「ストアバッファ」と呼ばれる）に一時的に格納するなどの対応で、逆依存関係あるいは出力依存関係を解消し、非プログラム順序実行を可能とする技術が、従来より、知られている。

【0018】

一方、正依存関係が存在する場合は、本質的に非プログラム順序で実行を行う

ことができず、プログラム順序で実行を行うことになる。

【0019】

しかしながら、ロード・ストア命令が対象とするアドレスは実行時まで不明である場合が多い。すなわち、デコード時に依存関係が判明するレジスタの場合と違い、メモリの場合には実行時まで依存関係が判明しない場合が多いため、非プログラム順序で実行を行えないことによる性能への影響が大きい。具体例を図9を参照して説明する。

【0020】

図9(a)は、ロード命令とストア命令からなるプログラム例のプログラム順序を示した図である。プログラム順に、

アドレスA1に対するロード命令LD1、
アドレスA4'に対するストア命令ST1、
アドレスA2に対するロード命令LD2、
アドレスA3に対するロード命令LD3、
アドレスA4に対するロード命令LD4、
を示している。

【0021】

ここで、ST1の対象アドレスA4'と、LD4の対象アドレスA4が等しい($A4' = A4$)とすると、ST1とLD4は同じアドレスを対象としており、かつ、LD4よりもST1の方がプログラム順序で前にあるため、ST1からLD4へ正依存関係が存在する。

【0022】

つまり、プログラムの意味上、ST1がアドレスA4' (= A4) に書き込みを行ったデータを、LD4がアドレスA4 (= A4') から読み出すことが期待されているため、それらの実行も、前にST1、その後にLD4のプログラム順序で行われる必要がある。

【0023】

図9(b)は、図9(a)で示したプログラム順序の命令列のプログラム順序実行の例を示す図である。

【0024】

左からサイクル番号、そのサイクルの実行命令、実行命令の対象アドレスを示す。ここで、ST1の対象アドレスA4'は、5サイクル目まで判明しないと想定すると、

1サイクル目にアドレスA1に対するロード命令LD1、
5サイクル目にアドレスA4'に対するストア命令ST1、
6サイクル目にアドレスA2に対するロード命令LD2、
7サイクル目にアドレスA3に対するロード命令LD3、
8サイクル目にアドレスA4に対するロード命令LD4、
が実行されることになる。

【0025】

仮に2サイクル目、または3サイクル目、または4サイクル目に、アドレスA2、またはアドレスA3、またはアドレスA4が判明していたとしても、LD2、またはLD3、またはLD4を、2サイクル目、または3サイクル目、または4サイクル目に、ST1を追い越して実行する非プログラム順序実行を行うことはできない。

【0026】

なぜなら、ST1の対象アドレスA4'が5サイクル目まで判明しないため、5サイクル目まで、ST1からLD1、LD2、LD3、LD4への正依存関係の有無も判明しないからである。

【0027】

すなわち、図9(b)に示したプログラム順序実行の例の場合、2サイクル目、3サイクル目、4サイクル目には、ロード・ストア命令を実行することができず、LD1、LD2、LD3、LD4、ST1の5命令の実行に8サイクルを要してしまう。

【0028】

このように、ロード・ストア命令に関して非プログラム順序実行を行わない方法では、実行性能が低下する、という問題があった。

【0029】

この問題に対して、正依存関係の有無が判明するよりも以前に、正依存関係が存在しないと仮定して、投機的に非プログラム順序実行を行ってしまう方式が、従来より、知られている。

【0030】

この明細書では、この方式による、データの正依存関係に対する投機的命令実行 (speculative execution) を、「データ依存投機実行」と表記する、ことにする。

【0031】

データ依存投機実行においては、

- ・実際に正依存関係が存在せず投機実行が成功する場合と、
- ・実際に正依存関係が存在して投機実行が失敗する場合と、

があり、正依存関係の有無が判明した時点で、そのいずれであるかを判定する必要がある。

【0032】

実際に正依存関係が存在せず投機実行が成功した場合、そのまま後続の命令実行を継続することが可能であり、データ依存投機実行による非プログラム順序実行を行った分だけ、実行性能の向上が期待できる。

【0033】

一方、実際に正依存関係が存在して投機実行が失敗した場合、プログラムの意味が変わってしまうため、プログラムの正しい実行結果が保証できなくなる。そのため、データ依存投機実行による非プログラム順序実行を行った命令の結果を取り消して、再びプログラム順序で再実行するなどのデータ依存投機実行の失敗に対する回復処理 (recovery) が必要となる。投機実行が失敗した場合、失敗した命令の取り消しや回復処理のために、プログラム順序実行するよりも性能が低下することが多い。しかし、投機実行に失敗する場合よりも成功する場合の確率が十分に高ければ、プログラム全体としての実行性能の向上が期待できることになる。

【0034】

なお、非プログラム順序実行については、マイク・ジョンソンによる文献1 (

“スーパースカラ・プロセッサ”、日経BP出版センター、1994年)に詳しい。

【0035】

また、投機実行の失敗による回復処理方法については、例えば、特開平5-224927号公報に開示されている方法がある。

【0036】

データ依存投機実行の具体例を、図9を参照して説明する。図9(c)は、図9(a)に示したプログラム順序の命令列に対して、データ依存投機実行を行いかつ投機実行に成功した場合の実行例を示す図である。

【0037】

図9(b)で示したプログラム順序実行の場合と同様に、ST1の対象アドレスA4'は5サイクル目まで判明しないとする。また、LD2の対象アドレスA2は2サイクル目で、LD3の対象アドレスA3は3サイクル目で、LD4の対象アドレスA4は6サイクル目で判明すると想定する。

【0038】

まず、1サイクル目にアドレスA1に対するロード命令LD1を実行する。

【0039】

次に2サイクル目において、プログラム順序ではST1を実行するのであるが、対象アドレスA4'がまだ不明であるために実行できない。このため、対象アドレスA2が判明しているLD2を、ST1を追い越して、非プログラム順序で実行する。ここで、ST1の対象アドレスA4'はまだ不明であるため、ST1からLD2への正依存関係の有無もまた不明である。すなわち、LD2はST1に対してデータ依存投機実行を行う。

【0040】

同様にして、3サイクル目ではST1はまだ実行できないため、対象アドレスA3が判明しているLD3のデータ依存投機実行を行う。このとき、ST1からLD3への正依存関係の有無もまた不明である。

【0041】

次に、4サイクル目では、ST1及びLD4ともにその対象アドレスは不明で

あるので、いずれの命令も実行できない。

【0042】

次に、5サイクル目においてST1の対象アドレスA4' が判明するので、ST1の実行を行う。同時に、ST1からデータ依存投機実行を行ったLD2、LD3に対する正依存関係の有無を判定する。この場合、LD2の対象アドレスA2、及びLD3の対象アドレスA3は、ST1の対象アドレスA4' と異なるため、LD2及びLD3に対して正依存関係は存在しない。

【0043】

すなわち、LD2及びLD3のデータ依存投機実行は成功したと判断する。従って、後続の命令実行をそのまま継続することができ、次の6サイクル目にLD4を実行する。LD4はST1から正依存関係があるが、プログラム順序で実行されているので問題は生じない。

【0044】

以上、データ依存投機実行が成功した場合について説明した。図9(b)に示したプログラム順序で実行した場合、実行に8サイクルを要したのに対して、データ依存投機実行による非プログラム順序実行が成功した場合、実行は6サイクルで完了するため、その分だけ実行性能が向上する。

【0045】

一方、図9(d)は、図9(a)に示したプログラム順序の命令列に対して、データ依存投機実行を行い、かつ、投機実行に失敗した場合の実行例を示している。図9(b)に示したプログラム順序実行の場合と同様に、ST1の対象アドレスA4' は5サイクル目まで判明しないとする。

【0046】

また、LD2の対象アドレスA2は2サイクル目で、LD3の対象アドレスA3は3サイクル目で、LD4の対象アドレスA4は4サイクル目で判明すると想定する。

【0047】

まず、1サイクル目にアドレスA1に対するロード命令LD1を実行する。

【0048】

次に、図9（c）に示したデータ依存投機実行が成功した場合と同様に、2サイクル目及び3サイクル目においては、プログラム順序で次の命令であるST1は、対象アドレスA4'がまだ不明であるために実行できないため、対象アドレスが判明しているLD2及びLD3をデータ依存投機実行による非プログラム順序実行を行う。

【0049】

次に、4サイクル目では、未だST1の対象アドレスA4'は不明であるが、LD4の対象アドレスA4は判明しているため、LD4のデータ依存投機実行を行う。

【0050】

次に、5サイクル目においてST1の対象アドレスA4'が判明するので、ST1の実行を行う。同時に、ST1からデータ依存投機実行を行ったLD2、LD3、LD4に対する正依存関係の有無を判定する。この場合、LD2の対象アドレスA2、及び、LD3の対象アドレスA3は、ST1の対象アドレスA4'と異なるため、LD2及びLD3に対しては、正依存関係は存在しない。

【0051】

しかし、LD4の対象アドレスA4は、ST1の対象アドレスA4'に等しいため、ST1からLD4に対して正依存関係が存在する。すなわち、正依存関係が存在するにもかかわらず、非プログラム順序で実行を行ってしまっているので、LD4のデータ依存投機実行は失敗したと判断する。

【0052】

この場合、このまま後続の命令の実行を継続すると、プログラムの正しい実行結果が保証されないので、データ依存投機実行の失敗の回復処理の実行が必要である。

【0053】

例えば、図9（d）に示した実行例では、5サイクル目においてデータ依存投機実行は失敗したと判断されると、まず、データ依存投機実行を行った2サイクル目から4サイクル目のLD2、LD3、LD4及び5サイクル目のST1の実行結果を取り消し、次に、実行結果を取り消したST1、LD2、LD3、LD

4 の再実行を 7 サイクル目からプログラム順序で実行することで、データ依存投機実行の失敗の回復を行い、その後、後続命令の実行を再開する。

【 0 0 5 4 】

この場合、LD 1、LD 2、LD 3、LD 4、ST 1 の 5 命令の実行に、1 0 サイクルを要することになり、プログラム順序で実行を行った場合の 8 サイクルより、性能が低下することになる。

【 0 0 5 5 】

しかしながら、データ依存投機実行に成功した場合には、6 サイクルで実行できるため、投機実行に失敗する場合よりも成功する場合の確率が十分に高ければ、プログラム全体としての実行性能の向上が期待できる。

【 0 0 5 6 】

以上述べたように、プロセッサがデータ依存投機実行を行うには、ロード・ストア命令間のメモリに対する正依存関係の有無を判定する機能が必要となる。この機能を担う装置として、例えば、Manoj Franklinらによる論文“ARB: A Hardware Mechanism for Dynamic Reordering of Memory References”, IEEE Transactions on Computers, vol.45, no.5(1996年5月)に開示されているデータ依存関係検出装置がある。

【 0 0 5 7 】

この従来のデータ依存関係検出装置の構成及び動作を図 1 0 を参照して説明する。図 1 0 は、従来のデータ依存関係検出装置 1 0 0 の構成を示すブロック図である。図 1 0 において、符号 1 0 1 はアドレスバッファ、符号 1 0 2 はアドレス比較器、符号 1 0 3 は論理和回路である。データ依存関係検出装置 1 0 0 は、ロード命令の対象アドレスを格納する複数のアドレスバッファ 1 0 1 と、各々のアドレスバッファに接続された複数のアドレス比較器 1 0 2 と、すべてのアドレス比較器 1 0 2 の出力を入力とする論理和回路 1 0 3 とから構成される。

【 0 0 5 8 】

アドレス比較器 1 0 2 はアドレスバッファ 1 0 1 に格納されたロード命令の対象アドレスと、実行中のストア命令の対象アドレスとの比較を行う。

【 0 0 5 9 】

論理和回路103は、すべてのアドレス比較器102の出力の論理和を演算し、データ依存検出結果として出力する。

【0060】

データ依存関係検出装置100によるストア命令からロード命令へのメモリに関する正依存関係の検出は、以下のような動作で実現される。まず、ロード命令がデータ依存投機実行された場合、その対象アドレスを、これまでロード命令の対象アドレスが格納されていない空いているアドレスバッファ101に格納していく。

【0061】

次に、ストア命令が実行されると、その対象アドレスは、すべてのアドレス比較器102に入力され、アドレスバッファ101に格納されている、それ以前にデータ依存投機実行されたロード命令の対象アドレスと比較する。

【0062】

アドレス比較器102から出力された比較結果はすべて論理和をとり、データ依存検出結果として出力する。ストア命令の対象アドレスと、アドレスバッファ101に格納されているロード命令の対象アドレスのいずれとも一致しない場合、そのストア命令からアドレスバッファ101に対象アドレスを格納したロード命令へ、正依存関係が存在しないと判断することができるので、その旨をデータ依存検出結果として出力する。この場合、データ依存投機実行が成功したとして、そのまま後続の命令の実行を継続することができる。

【0063】

一方、ストア命令の対象アドレスと、アドレスバッファ101に格納されているロード命令の対象アドレスのいずれかが一致した場合、そのストア命令からアドレスバッファ101に対象アドレスを格納したロード命令へ、正依存関係が存在すると判断することができるので、その旨をデータ依存検出結果として出力する。この場合、データ依存投機実行が失敗したとして、データ依存投機実行が失敗したことによる回復処理などが行われることになる。

【0064】

上述した従来のデータ依存関係検出装置100には、次のような問題点がある

【0065】

第1の問題点は、ハードウェア量が大きいことである。データ依存投機実行においてプログラムの実行結果が正しく得られることを保証するには、すべての正依存関係をもれなく検出することが必要である。

【0066】

従来のデータ依存関係検出装置100では、データ依存投機実行を行ったすべてのロード命令の対象アドレスをアドレスバッファ101に格納し、アドレス比較器102により、後続のストア命令の対象アドレスと比較する必要がある。そのため、すべてのアドレスバッファ101に、それまでデータ依存投機実行したロード命令の対象アドレスが既に格納されて、空いているアドレスバッファ101がない場合には、それ以上ロード命令をデータ依存投機実行することができない。

【0067】

この場合、それ以降のロード・ストア命令はプログラム順序で実行されることになるため、データ依存投機実行することが可能なロード命令数は、アドレスバッファ101の設置数及びそれと同数のアドレス比較器102の設置数に制限されることになる。従って、データ依存投機実行による実行性能の向上を享受するためには、十分な数のアドレスバッファ101及びアドレス比較器102を設置する必要があるため、必要なハードウェア量が大きくなってしまいう問題がある。

【0068】

第2の問題点は、正依存関係の検出処理の処理速度が遅いことである。従来のデータ依存関係検出装置100では、正依存関係の検出処理に、アドレス比較器102によるアドレス比較の処理時間に加えて、論理和回路103によるすべてのアドレス比較器102からの出力の論理和の演算時間が必要なため、動作速度が遅く、その結果、プロセッサの動作周波数を高速化することが困難であるという問題点がある。

【0069】

この問題は、アドレスバッファ 1 0 1 及びアドレス比較器 1 0 2 の設置数が多くなるにつれて、論理和回路 1 0 3 の入力数が増加するため、益々顕著になる。すなわち、第 1 の問題点で述べたように、従来のデータ依存関係検出装置 1 0 0 では、データ依存投機実行による実行性能の向上を享受するために、必要なハードウェア量が大きくなってしまいが、その場合、正依存関係検出処理の処理時間が益々増加し、プロセッサの動作周波数を高速化することが困難になるため、結局は実行性能が低下してしまうという問題がある。

【 0 0 7 0 】

【発明が解決しようとする課題】

本発明は、このような事情に鑑みてなされたもので、その目的は、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されず、小さいハードウェア量でもデータ依存投機実行による実行性能の向上を享受することのできる、ハードウェア量が小さいデータ依存関係検出装置を提供することにある。

【 0 0 7 1 】

また、本発明の別の目的は、正依存関係の検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易である、高速なデータ依存関係検出装置を提供することにある。

【 0 0 7 2 】

【課題を解決するための手段】

上記目的を達成するために、本発明は、プロセッサが非プログラム順序でロード命令またはストア命令の実行を行う際に、前記ロード命令またはストア命令が処理対象とするアドレス間の依存関係の存在の可能性を検出するデータ依存関係検出装置であって、実際に前記依存関係が存在する場合は必ず依存関係が存在する旨を検出するが、実際に前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容する構成としたものであり、ロード命令及びストア命令が処理対象とするアドレスを一意的番号に変換するアドレス変換手段と、前記変換された番号に対応して、ロード命令又はストア命令が実行されたか否かを記憶する複数の記憶手段から構成される実行履歴記憶手段とを少なくと

も含むことを特徴とする。

【 0 0 7 3 】

上記構成によれば、実行されたロード命令またはストア命令が処理対象とするアドレス自体をすべて記憶しておく必要がなく、また前記アドレスを比較するためのアドレス比較器も必要としない。

【 0 0 7 4 】

すなわち、本発明のデータ依存関係検出装置によれば、データ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。また、ロード命令、ストア命令の対象アドレス間の依存関係の検出処理の際に、ロード命令またはストア命令の対象アドレスの比較を必要しない。さらに、前記実行履歴手段を構成する複数の記憶手段のうち、読み出しの対象となる記憶手段は高々1つだけで充分なので、依存関係の検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【 0 0 7 5 】

また、本発明は、ロード命令及びストア命令などのメモリ操作命令を実行可能な複数のプロセッサから構成されるマルチプロセッサシステムにおいて、前記各々のプロセッサが備えるデータ依存関係検出装置であって、かつ前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むロード命令またはストア命令が処理対象とするアドレスと、自プロセッサが実行するスレッドが含むロード命令またはストア命令が処理対象とするアドレスとの間に依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置であって、前記データ依存関係検出装置は、実際に前記依存関係が存在する場合は必ず依存関係が存在する旨を検出するが、実際に前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容することを特徴とし、自プロセッサ及び他プロセッサが実行するロード命令及び

ストア命令が処理対象とするアドレスを一意な番号に変換する複数のアドレス変換手段と、前記変換された番号に対応して、ロード命令又はストア命令が実行されたか否かを記憶する複数の記憶手段から構成される実行履歴記憶手段とを少なくとも含むことを特徴とする。

【 0 0 7 6 】

上記構成によれば、スレッド並列処理におけるスレッド単位のデータ依存投機実行においても、データ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、データ依存投機実行を行うことのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもスレッド単位のデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。さらに、スレッド間の依存関係の検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【 0 0 7 7 】

また、本発明によるデータ依存関係検出装置は、ロード命令及びストア命令などのメモリ操作命令を実行可能な複数のプロセッサから構成されるマルチプロセッサシステムにおいて、前記各々のプロセッサが備えるデータ依存関係検出装置であって、前記マルチプロセッサシステムがスレッド単位で並列処理を行う際に、他のプロセッサが実行するスレッドが含むロード命令またはストア命令が処理対象とするアドレスと、自プロセッサが実行するスレッドが含むロード命令またはストア命令が処理対象とするアドレスとの間に依存関係が存在する可能性を検出する機能、及び自プロセッサが実行するスレッドが含むロード命令またはストア命令が対象とするアドレス間の依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置であって、前記データ依存関係検出装置は、実際に前記依存関係が存在する場合は必ず依存関係が存在する旨を検出するが、実際に前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容することを特徴とし、自プロセッサ及び他プロセッサが実行するロード命令及びストア命令が処理対象とするアドレスを一意な番号に変換する複数のアドレス変換手段と、前記変換された番号に対応して、ロード命令又はストア命

令が実行されたか否かを記憶する複数の記憶手段から構成される実行履歴記憶手段とを少なくとも含むことを特徴とする。

【 0 0 7 8 】

上記構成によれば、スレッド並列処理におけるスレッド単位のデータ依存投機実行及び、スレッド内の命令単位のデータ依存投機実行を可能とし、さらにそれらデータ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもスレッド単位のデータ依存投機実行及びスレッド内の命令単位のデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。さらに、データ依存関係の検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【 0 0 7 9 】

【発明の実施の形態】

次に、本発明の実施の形態について図面を参照して詳細に説明する。

【 0 0 8 0 】

図 1、は本発明の第 1 の実施の形態のデータ依存関係検出装置を含むプロセッサの構成を示すブロック図である。図 1 において、符号 1 0 は本発明の第 1 の実施の形態によるデータ依存関係検出装置、符号 1 7 はプロセッサ制御部、符号 1 8 は命令実行部である。プロセッサの他の構成要素については図示していない。

【 0 0 8 1 】

プロセッサが命令を実行する場合、プロセッサ制御部 1 7 は、実行する命令の種別を命令実行部 1 8、及び、データ依存関係検出装置 1 0 に出力する。

【 0 0 8 2 】

データ依存関係検出装置 1 0 は、その実行する命令種別を実行命令 1 4 から入力する。

【 0 0 8 3 】

さらに、実行する命令がロード命令またはストア命令の場合、プロセッサ制御

部 17 は、その対象アドレスを命令実行部 18 及びデータ依存関係検出装置 10 に出力する。

【0084】

データ依存関係検出装置 10 は、そのロード・ストア命令の対象アドレスをロード・ストアアドレス 15 から入力する。

【0085】

さらに、ロード命令またはストア命令をデータ依存投機実行する場合、プロセッサ制御部 17 は、その旨を投機実行フラグ 13 を通じてデータ依存関係検出装置 10 に通知する。

【0086】

データ依存関係検出装置 10 は、ハッシュ関数回路 11、及び命令履歴テーブル 12 を備えており、プロセッサ制御部 17 からの入力として、投機実行フラグ 13、実行命令 14、ロード・ストアアドレス 15 を有し、プロセッサ制御部 17 への出力としてデータ依存検出結果 16 を有する。

【0087】

ハッシュ関数回路 11 は、 m ビットで表現されるロード命令またはストア命令の対象アドレスを、 n ビットで表現される命令履歴テーブル 12 のエントリ番号に変換するハッシュ関数 f を実現する論理回路である。通常、 m よりも n が小さい ($m > n$)。ここで、ハッシュ関数 f は、同一の入力に対しては、同一の値をもつ。すなわち、 $N1 = f(A1)$ かつ $N2 = f(A2)$ としたとき、 $A1 = A2$ であれば、 $N1 = N2$ が成立する関数である。従って、ハッシュ関数回路 11 は、同一のロード・ストア命令の対象アドレスに対しては、必ず同一の命令履歴テーブルのエントリ番号を出力することを保証する。

【0088】

一方、 $A1 \neq A2$ であっても、一般には $N1 \neq N2$ は保証されない。すなわち、異なるロード・ストア命令の対象アドレスに対して、同一の命令履歴テーブルのエントリ番号が出力される場合が存在する。

【0089】

命令履歴テーブル 12 は、1 ビットの情報を格納できる複数のエントリから構

成される。各エントリはメモリやフリップフロップなどの記憶手段で実現することができる。命令履歴テーブル 1 2 が備える上記エントリの総数は、ハッシュ関数回路 1 1 の出力が n ビット幅である場合、2 の n 乗に等しい数である。命令履歴テーブル 1 2 は、ハッシュ関数回路 1 1 が出力するエントリ番号が指し示すエントリに対して書き込み及び読み出し処理を行う。

【 0 0 9 0 】

さらに、プロセッサ制御部 1 7 からデータ依存関係検出装置 1 0 に入力される投機実行フラグ 1 3 に基づいて、すべてのエントリに同一の値を書き込む初期化機能を有する。

【 0 0 9 1 】

上記構成のデータ依存関係検出装置 1 0 の動作について、図 1 及び図 2 を参照して説明する。

【 0 0 9 2 】

図 2 は、データ依存関係検出装置 1 0 の動作を示すタイミング図である。まず、初期状態においては、命令履歴テーブル 1 2 を構成するすべてのエントリは、ロード命令が実行されていないことを示す第 1 の状態（例えば論理 0）にある。プロセッサがデータ依存関係に対して投機的な命令実行を行わず、プログラム順序で確定的に命令実行を行う状態にある場合（この状態を「確定実行状態」という）、データ依存関係検出装置 1 0 はなんら動作を行わない。図 2 においては、サイクル 0 からサイクル 3 までが確定実行状態である。

【 0 0 9 3 】

それに対して、プロセッサがデータ依存関係に対して投機的な命令実行を行う状態（この状態を「投機実行状態」という）にある場合、データ依存関係検出装置 1 0 は、ストア命令からロード命令への正依存関係の検出処理を行う。図 2 においては、サイクル 4 から 8 までが投機実行状態である。

【 0 0 9 4 】

プロセッサが確定実行状態または投機実行状態のどちらにあるかは、プロセッサ制御部 1 7 からデータ依存関係検出装置 1 0 に入力される投機実行フラグ 1 3 で判断される。図 2 においては、投機実行フラグ 1 3 の値が論理 0 である場合、

プロセッサは確定実行状態にあることを示し、一方、投機実行フラグ13の値が論理1である場合、プロセッサは投機実行状態にあることを示す。

【0095】

投機実行状態において、投機的なロード命令が実行されると（図2のサイクル5）、データ依存関係検出装置10は、投機的なロード命令が実行されたことを命令履歴テーブル12に記憶する。より詳細には、ロード命令の対象アドレスをA1とすると、その対象アドレスA1をハッシュ関数回路11によりエントリ番号N1に変換し、命令履歴テーブル12に入力する。命令履歴テーブル12は、入力されたエントリ番号N1に対応するエントリを、投機的なロード命令が実行されたことを示す第2の状態（例えば論理1）に変更する。

【0096】

一方、投機的なストア命令が実行された場合（図2のサイクル7）には、データ依存関係検出装置10の対応するエントリに記憶された状態の読み出しを行う。より詳細には、ストア命令の対象アドレスをA2とすると、その対象アドレスA2をハッシュ関数回路11によりエントリ番号N2に変換し、命令履歴テーブル12に入力する。命令履歴テーブル12は、入力されたエントリ番号N2に対応するエントリの内容を読み出し、その内容に基づいてデータ依存検出結果16からプロセッサ制御部17に出力する。

【0097】

例えば図2において、サイクル5で実行されたロード命令の対象アドレスA1と、サイクル7で実行されたストア命令の対象アドレスA2が等しい場合（ $A1 = A2$ ）、ハッシュ関数回路11の性質により、それぞれに対応するエントリ番号N1、N2は互いに等しい（ $N1 = N2$ ）。すなわち、サイクル5にロード命令によって命令履歴テーブル12のエントリN1に格納された値（論理1）は、サイクル7で実行されたストア命令によって命令履歴テーブル12から読み出されるため、サイクル7で実行されたストア命令からサイクル5で実行されたロード命令へ、正依存関係が存在する可能性があることが検出される。

【0098】

ここで検出されるのが、“正依存関係が存在すること”ではなく、“正依存関

係が存在する可能性があること”であるのは、後述するエイリアスの発生する場合があるために、ストア命令によって命令履歴テーブル 1 2 から投機的なロード命令が実行されたことを示す値が読み出された場合でも、実際には正依存関係が存在しない場合があるためである。

【 0 0 9 9 】

いずれにせよ、正依存関係が存在する可能性があることが検出された場合、データ依存関係検出装置 1 0 は、データ依存投機実行が失敗したことを示す値（例えば論理 1）をデータ依存検出結果 1 6 に出力する。その場合、プロセッサ制御部 1 7 及び命令実行部 1 8 はデータ依存投機実行が失敗したことに對する回復処理を実行する。

【 0 1 0 0 】

一方、サイクル 5 で実行されたロード命令の対象アドレス A 1 と、サイクル 7 で実行されたストア命令の対象アドレス A 2 が異なる場合（ $A 1 \neq A 2$ ）、ハッシュ関数回路 1 1 の性質により、それぞれに對應するエントリ番号 N 1、N 2 は互いに異なる場合（ $N 1 \neq N 2$ ）と、互いに等しい場合（ $N 1 = N 2$ ）の両方が存在する。前者の、エントリ番号 N 1、N 2 が互いに異なる場合、サイクル 7 のストア命令実行時には、命令履歴テーブル 1 2 のエントリ N 2 の値は初期値である論理 0 であるため、サイクル 7 で実行されたストア命令からサイクル 5 で実行されたロード命令へ、正依存関係が存在しないことが検出される。この場合、データ依存投機実行が成功したことを示す値（例えば論理 0）をデータ依存検出結果 1 6 に出力する。プロセッサ制御部 1 7 はデータ依存投機実行が成功したとして、データ依存投機実行が失敗したことに對する回復処理は実行せず、後続の命令実行をそのまま続行することができる。

【 0 1 0 1 】

後者の、エントリ番号 N 1、N 2 が互いに等しい場合、サイクル 5 のロード命令とサイクル 7 のストア命令は、同じ命令履歴テーブル 1 2 のエントリ N 1（ $= N 2$ ）を参照する。すなわち、ロード命令・ストア命令それぞれの対象アドレス A 1、A 2 が互いに異なり、ストア命令からロード命令へ正依存関係が存在しないのにもかかわらず、サイクル 7 のストア命令実行時には、サイクル 5 でロード

命令により書き込まれた論理 1 が命令履歴テーブル 1 2 から読み出されるため、サイクル 7 で実行されたストア命令からサイクル 5 で実行されたロード命令へ、正依存関係が存在する可能性があることが検出される。この場合、データ依存投機実行が失敗したことを示す値（例えば論理 1）をデータ依存検出結果 1 6 に出力する。プロセッサ制御部 1 7 及び命令実行部 1 8 は、データ依存投機実行が失敗したことに対する回復処理を実行する。上記したように、互いに異なる対象アドレスが、命令履歴テーブル 1 2 の同じエントリに割り当てられることを、以下では“エイリアスが発生する”と表記する。

【 0 1 0 2 】

ロード命令とストア命令の間にエイリアスが発生した場合、正依存関係が存在しない場合であっても、プロセッサ制御部 1 7 には正依存関係が存在した旨が通知され、本来不必要なデータ依存投機実行失敗による回復処理が実行されてしまう。しかし、この場合でも回復処理は不必要ではあるが、プログラム上の意味を変更してしまうことはないため、プログラムの実行結果の正しさは保証される。

【 0 1 0 3 】

一方で、不必要な回復処理が実行されることにより、プログラムの実行性能が低下するという問題がある。しかし、この問題に対しては後述するように、ハッシュ関数回路 1 1 のハッシュ関数を適切に選択したり、命令履歴テーブル 1 2 のエントリを十分多く実装することにより、エイリアスの発生確率は低減することができる。エイリアスの発生確率が充分小さければ、データ依存投機実行の効果により、プログラム全体の実行性能の向上を期待することができる。

【 0 1 0 4 】

エイリアスはロード命令の間にも発生する。例えば、対象アドレスが A 1 であるロード命令 L D 1、及び対象アドレスが A 2 であるロード命令 L D 2 の 2 つのロード命令が実行されたとき、それぞれの対象アドレスが異なっている（ $A 1 \neq A 2$ ）場合でも、参照されるそれぞれの命令履歴テーブル 1 2 のエントリ N 1 及び N 2 が互いに同じになる（ $N 1 = N 2$ ）エイリアスが発生する場合がある。この場合、命令履歴テーブル 1 2 の同じエントリ N 1（= N 2）にロード命令が実行されたことを示す論理 1 が格納される。

【 0 1 0 5 】

次に、LD 1 または LD 2 に正依存関係のある対象アドレスが A 1 または A 2 のストア命令が実行された場合、命令履歴テーブル 1 2 のエントリ N 1 (= N 2) が参照され、正依存関係の存在が検出されるが、LD 1、LD 2 のどちらに対して正依存関係が存在するかは区別することができない。

【 0 1 0 6 】

しかし、正依存関係が存在すれば必ず検出され、データ依存投機実行失敗による回復処理が実行されるため、この場合も、プログラムの実行結果の正しさは保証される。

【 0 1 0 7 】

上述したように、本発明によるデータ依存関係検出装置は、ストア命令からロード命令への正依存関係の存在を厳密に検出するのではなく、ストア命令からロード命令への正依存関係の存在の可能性を検出する、ことを主たる特徴の一つとしている。

【 0 1 0 8 】

任意のロード命令の対象アドレスは、ハッシュ関数回路 1 1 により、命令履歴テーブル 1 2 の特定のエントリに重複を許して割り当てるため、従来のデータ依存関係検出装置と異なり、実行されたロード命令の対象アドレス自体をすべて記憶しておく必要がなく、またストア命令の対象アドレスと比較するためのアドレス比較器も必要としない。

【 0 1 0 9 】

すなわち、この実施の形態によるデータ依存関係検出装置によれば、データ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、この実施の形態によるデータ依存関係検出装置によれば、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。

【 0 1 1 0 】

また、この実施の形態によるデータ依存関係検出装置によれば、ストア命令の実行時の正依存関係の検出処理において、ストア命令の対象アドレスとロード命令の対象アドレスの比較を必要しない。

【0111】

さらに、命令履歴テーブルの読み出すエントリは1つだけなので、すべてのエントリ出力の論理和を演算する必要もない。すなわち、この実施の形態によるデータ依存関係検出装置によれば、正依存関係の検出の処理時間が短いため、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【0112】

次に、本発明の第1の実施の形態によるデータ依存関係検出装置の動作の具体例を説明する。

【0113】

図3は、ハッシュ関数回路21及び命令履歴テーブル22から構成されるデータ依存関係検出装置20の構成を模式的に示すブロック図である。図3のハッシュ関数回路21、命令履歴テーブル22、データ依存関係検出装置20は、図1のハッシュ関数回路11、命令履歴テーブル12、データ依存関係検出装置10にそれぞれ対応している。実行命令24、投機実行フラグ23、データ依存検出結果26は、図1の実行命令14、投機実行フラグ13、データ依存検出結果16にそれぞれ対応している。プロセッサ制御部（図3では図示せず、図1の17）からの入力として、投機実行フラグ、実行命令、ロード・ストアアドレスを備える。また、プロセッサ制御部への出力として、データ依存検出結果を備える。ここで、ロード・ストア命令の対象アドレスは8ビットで表現されるものとする。

【0114】

ハッシュ関数回路21の第1の実施例を、図5(a)に示す。図5(a)を参照すると、8ビットで与えられるロード・ストア命令の対象アドレスの2ビット目及び3ビット目をそのまま取り出すことで、2ビットで表現される命令履歴テーブル22のエントリ番号に変換する。命令履歴テーブル22は、4つのエントリ

りから構成され、ハッシュ関数回路 2 1 が出力する 2 ビット表現のエントリ番号、及び、プロセッサ制御部が出力する実行命令と投機実行フラグを参照して、各エントリの書き込みや読み出し処理を行う。

【 0 1 1 5 】

図 4 を参照して、データ依存関係検出装置 2 0 の動作の具体例を以下に説明する。まず、図 4 (a) はプログラム例における命令のプログラム順序を示しており、

ロード命令 L D 1、
ストア命令 S T 1、
ロード命令 L D 2、
ロード命令 L D 3、
ロード命令 L D 4

の順序である。ここで、L D 1 の対象アドレスは A 1、L D 2 の対象アドレスは A 2、L D 3 の対象アドレスは A 3、L D 4 の対象アドレスは A 4、S T 1 の対象アドレスは A 4 である。

【 0 1 1 6 】

L D 4 と S T 1 の対象アドレスは同じ A 4 であり、プログラム順序で S T 1 の方が前にあるため、S T 1 から L D 4 にはデータ依存関係が存在する。すなわち、実行時において L D 4 よりも S T 1 が前に実行されて、S T 1 がアドレス A 4 に書き込みを行った結果を、L D 1 がアドレス A 4 から読み出しを行わなければ、プログラムの実行結果の正しさは保証されない。

【 0 1 1 7 】

プロセッサがデータ依存投機実行状態にある場合、ストア命令からロード命令への正依存関係の有無が不明であっても、実行可能になった命令から実行される。

【 0 1 1 8 】

図 4 (b) は、データ依存投機実行状態における、図 4 (a) に示したプログラム順序の命令列のデータ依存投機実行による非プログラム順序実行例であり、左から順にサイクル番号、そのサイクルの実行命令、その実行命令の対象アドレ

スをそれぞれ示す。すなわち、0サイクル目にLD1、1サイクル目にLD2、2サイクル目にLD2、3サイクル目にLD3、4サイクル目にLD4、5サイクル目にST1、が実行されている。各実行命令の対象アドレスは、2進数8ビットで表記しており、A1=00100000、A2=00101000、A3=00001100、A4=00010000、である。

【0119】

正依存関係にあるST1及びLD4について、5サイクル目のST1よりも後に実行されるべきLD4が、それより前の4サイクル目に実行されているため、正依存関係に違反しており、データ依存投機実行に失敗している。すなわち、LD4はアドレスA4から、ST1がアドレスA4に書き込みを行った値ではなく、ST1が書き込みを行う以前の値を読み出してしまうため、プログラムの正しい実行結果を得ることが保証されないことになる。

【0120】

データ依存関係検出装置20（図3参照）は、このデータ依存投機実行の失敗を検出し、プロセッサ制御部17（図1参照）へデータ依存検出結果16（図1参照）の出力によって通知する機能を担う。この通知により、プロセッサ制御部は、正しいプログラムの実行結果が選られるように、データ依存投機実行した命令の再実行などのデータ依存投機実行の失敗による回復処理を行う。

【0121】

図4（c）は、図4（b）に示した命令実行順序に対するデータ依存関係検出装置20の動作を示す図である。左から順にサイクル番号、そのサイクルの投機実行フラグ（図1の13）の値、そのサイクルに実行されたロード・ストア命令が参照する命令履歴テーブル22のエントリ番号、そのサイクルの命令履歴テーブル22の状態をそれぞれ示す。命令履歴テーブル22の状態としては、図の右から順に、エントリ00、エントリ01、エントリ10、及び、エントリ11の4つのエントリの内容が示されている。

【0122】

例えば、0サイクル目は投機実行フラグの値は論理0であり、またロード・ストア命令は実行されていない。命令履歴テーブル22は初期状態にあり、4つの

エントリの内容はすべて、ロード命令が実行されていないことを示す論理0である。

【0123】

次に1サイクル目から5サイクル目においてプロセッサは投機実行状態になり、その旨を示す投機実行フラグが論理1になる。

【0124】

1サイクル目では、ロード命令LD1がデータ依存投機実行される。ハッシュ関数回路21は、LD1の対象アドレス $A1 = 00100000$ の2ビット目及び3ビット目から、命令履歴テーブル22に参照するエントリの番号00を出力する。命令履歴テーブル22は、ハッシュ関数回路21が出力するエントリ番号00、及び、プロセッサ制御部が出力する実行命令と投機実行フラグを参照して、エントリ00の内容を、ロード命令が実行されたことを示す論理1にする。

【0125】

次に2サイクル目において、ロード命令LD2がデータ依存投機実行される。ハッシュ関数回路21は、LD2の対象アドレス $A1 = 00101000$ の2ビット目及び3ビット目から、命令履歴テーブル22に対して、参照エントリの番号10を出力する。命令履歴テーブル22は、ハッシュ関数回路21が出力するエントリ番号10、及び、プロセッサ制御部が出力する実行命令と投機実行フラグを参照して、エントリ10の内容をロード命令が実行されたことを示す論理1を書き込む。

【0126】

同様にして、3サイクル目においては、ロード命令LD3がデータ依存投機実行され、その対象アドレスが $A3 = 00001100$ であることから、命令履歴テーブル22のエントリ11に論理1を書き込む。

【0127】

4サイクル目においては、ロード命令LD4がデータ依存投機実行され、その対象アドレスが $A4 = 00010000$ であることから、命令履歴テーブル22のエントリ00に論理1を書き込む。

【0128】

5 サイクル目においては、ストア命令 S T 1 がデータ依存投機実行され、その対象アドレスが A 4 = 0 0 0 1 0 0 0 0 であることから、命令履歴テーブル 2 2 のエントリ 0 0 の内容が読み出される。

【 0 1 2 9 】

命令履歴テーブル 2 2 のエントリ 0 0 には、1 サイクル目と 4 サイクル目に論理 1 が書き込まれているため、エントリ 0 0 からは論理 1 が読み出される。すなわち、ストア命令 S T 1 から、ロード命令 L D 1 または L D 4 への正依存関係が検出される。

【 0 1 3 0 】

データ依存投機実行状態にあるため、データ依存関係検出装置 2 0 は、データ依存検出結果 2 6 を論理 1 とし、プロセッサ制御部（図 1 の 1 7）に対して、データ依存投機実行が失敗したことを通知する。

【 0 1 3 1 】

データ依存投機実行の失敗により、次の 6 サイクル目からプロセッサ制御部（図 1 の 1 7）は、データ依存投機状態中の実行命令の結果の破棄や命令の再実行等、データ依存投機実行の失敗による回復処理を行う。また、データ依存投機実行が失敗により終了して、投機実行フラグ 2 3 が再び論理 0 に戻るため、命令履歴テーブル 2 2 は、すべてのエントリの内容を初期状態である論理 0 でクリアして、次のデータ依存投機実行に備える。

【 0 1 3 2 】

以上、本発明の第 1 の実施の形態によるデータ依存関係検出装置における、データ依存投機実行が失敗した場合の動作の具体例を説明した。一方、正依存関係が検出されずデータ依存投機実行が成功した場合には、データ依存投機実行の失敗による回復処理は行わず、そのまま後続の命令の実行を継続することができる。

【 0 1 3 3 】

図 4（c）に示したデータ依存関係検出装置 2 0 の動作において、L D 1 の対象アドレス A 1 = 0 0 1 0 0 0 0 0 と、L D 4 の対象アドレス A 4 = 0 0 0 1 0 0 0 0 は互いに異なるのににもかかわらず、命令履歴テーブル 2 2 の同じエントリ

00が参照されるエイリアスが発生している。

【0134】

これは、ハッシュ関数回路21がアドレスA1及びアドレスA4を同じエントリ番号00に変換していることが原因である。従って、仮に、ストア命令ST1が実行されるより以前にロード命令LD4が実行されず、ストア命令ST1からロード命令LD4に対して正依存関係が検出されなかった場合であっても、ストア命令ST1からロード命令LD1に対して、偽の正依存関係が検出されてしまうため、データ依存投機実行は成功しているにもかかわらず、失敗と判断されてしまう。そのため、本来実行する必要のないデータ依存投機実行失敗による回復処理が実行されてしまう。データ依存投機実行失敗による回復処理は、プログラム上の意味を変更してしまうことはないため、エイリアスが発生した場合でも、プログラムの実行結果の正しさは保証される。

【0135】

しかし、このように、エイリアスが発生すると、不必要な回復処理が実行されるので、プログラムの実行性能が低下するという問題がある。このエイリアスによる性能低下の問題に対しては、ハッシュ関数回路21のハッシュ関数を適切に選択したり、命令履歴テーブル22のエントリを十分多く実装することにより、エイリアスの発生確率を十分小さくすることで低減が可能である。

【0136】

例えば、命令履歴テーブル22のエントリ数を、4から8に増加し、ハッシュ関数回路21にアドレスの2ビット目、3ビット目、4ビット目を取り出す第2の実施例（図5（b））を使用した場合、アドレスA1=00100000はエントリ000へ、アドレスA2=00101000はエントリ010へ、アドレスA3=00001100はエントリ011へ、アドレスA4=00010000はエントリ100へ割り当てられる。すなわち、エイリアスが発生しない。

【0137】

あるいは、命令履歴テーブル22のエントリ数は4のままである場合でも、ハッシュ関数回路21に、出力するエントリ番号の0ビット目を入力されるアドレスの2ビット目と4ビット目の排他的論理和、出力するエントリ番号の1ビット

目を入力されるアドレスの3ビット目と5ビット目の排他的論理和とする第3の実施例（図5（c）参照）を使用すると、アドレスA1=00100000はエントリ10へ、アドレスA2=00101000はエントリ00へ、アドレスA3=00001100はエントリ11へ、アドレスA4=00010000はエントリ01へ割り当てられる。すなわち、エイリアスが発生しない。

【0138】

このように、エイリアスの発生確率は低減させることが可能であり、その場合、データ依存投機実行の効果により、プログラム全体の実行性能の向上を期待することができる。

【0139】

以上説明したように、本発明によるデータ依存関係検出装置によれば、データ依存投機実行の成否判定に必要なハードウェア量が小さいという効果がある。これは、従来のデータ依存関係検出装置と異なり、実行されたロード命令の対象アドレス自体をすべて記憶しておく必要がなく、またストア命令の対象アドレスと比較するためのアドレス比較器も必要としないことによる。

【0140】

さらには、エイリアスの存在を容認することにより、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもデータ依存投機実行による実行性能の向上を享受することができるためである。

【0141】

本発明の第1の実施の形態によるデータ依存関係検出装置による別の作用効果としては、データ依存投機実行の成否判定が高速に行えるため、プロセッサの動作周波数を高速化することが容易であることである。これは、ストア命令の実行時の正依存関係の検出処理において、ストア命令の対象アドレスとロード命令の対象アドレスの比較を必要とせず、また、命令履歴テーブルの読み出すエントリは1つだけなので、すべてのエントリ出力の論理和を演算する必要もないことによる。

【0142】

上記した本発明の第 1 の実施の形態は、ロード命令が実行された場合には、該ロード命令が処理対象とするアドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリに、ロード命令が実行されたことを示す値を格納し、ストア命令が実行された場合には、該ストア命令が処理対象とするアドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリからロード命令が実行されたか否かを示す値を読み出すことで、ストア命令からロード命令への正依存関係が存在する可能性を検出することを特徴としている。しかしながら、この実施の形態は、正依存関係が存在する可能性の検出にとどまらず、逆依存関係や出力依存関係が存在する可能性の検出に対しても、同様の構成で容易に適用することができる。

【 0 1 4 3 】

例えば、ストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリに、ストア命令が実行されたことを示す値を格納し、ロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリからストア命令が実行されたか否かを示す値を読み出すことで、ロード命令からストア命令への逆依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置が実現される。

【 0 1 4 4 】

または、ストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリからストア命令が実行されたか否かを示す値を読み出すのに加えて、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリに、ストア命令が実行されたことを示す値を格納することで、ストア命令間の出力依存関係が存在する可能性

を検出する機能を有するデータ依存関係検出装置が実現される。

【 0 1 4 5 】

さらに、ロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリからストア命令が実行されたか否かを示す値を読み出すのに加えて、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリに、ロード命令が実行されたことを示す値を格納し、ストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 1 1 により命令履歴テーブル 1 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリからロード命令が実行されたか否かを示す値及びストア命令が実行されたか否かを示す値を読み出すのに加えて、前記変換されたエントリ番号に対応する命令履歴テーブル 1 2 のエントリに、ストア命令が実行されたことを示す値を格納することで、ストア命令からロード命令への正依存関係が存在する可能性、ロード命令からストア命令への逆依存関係が存在する可能性、及びストア命令間の出力依存関係が存在する可能性、のすべてを検出する機能を有するデータ依存関係検出装置が実現される。

【 0 1 4 6 】

次に、本発明の第 2 の実施の形態におけるデータ依存関係検出装置について説明する。本発明の第 2 の実施の形態におけるデータ依存関係検出装置は、マルチプロセッサシステムにおいて、スレッド並列処理におけるデータ依存投機実行を実現する。ここでスレッド並列処理とは、単一のプログラムあるいは複数のプログラムをスレッドと呼ぶ複数のプログラムの部分単位に分割し、それぞれのスレッドを複数のプロセッサなどで並列に処理することでプログラム実行を高速化する、プログラム処理の高速化方法の一つである。

【 0 1 4 7 】

スレッドは、プログラムの一部分単位であり、複数の命令からなる命令列なので、複数のスレッド間で順序関係が定義される場合がある。例えば、同一のプログラムに属する 2 つのスレッドをそれぞれ T 0、T 1 としたとき、T 0 と T 1 の

間には、そのプログラム中の位置関係により、明確に順序関係が存在する。

【0148】

仮に、プログラムにおいてT0がT1よりも前に位置しているならば、プログラムの意味上、T0がT1よりも前に実行されることを期待されていることになる。逆に、プログラムにおいてT1がT0よりも前に位置しているならば、プログラムの意味上、T1がT0よりも前に実行されることを期待されていることになる。しかし、T0及びT1に含まれる命令間に依存関係が存在しない場合、T0とT1間の順序関係を無視した順序で実行しても、プログラムの意味は変わらないため、正しいプログラムの実行結果を得ることができる。

【0149】

スレッド並列処理では、上記した性質を利用して、プログラムの正しい実行結果を保証しつつ、スレッド間の順序関係によらずにスレッドの実行を並列に行うことで、プログラム実行を高速化する。命令単位の非プログラム順序実行と対比して、スレッド単位の非プログラム順序実行と位置づけられる。

【0150】

スレッド単位の非プログラム順序実行においても、正しいプログラムの実行結果を得るためには、スレッドが含む命令間の依存関係の解消や保証が必要である。

【0151】

しかしながら、命令単位の非プログラム順序実行と同様に、特にメモリに関する正依存関係に対しては、本質的にプログラム順序で実行する必要がある、かつ、確定的にプログラム順序実行を行うと、非プログラム順序実行の実行性能向上の効果が十分得られないという問題がある。

【0152】

特に、スレッド単位の非プログラム順序実行では、複数の命令からなるスレッド単位で、非プログラム順序実行が妨げられるので、より問題は深刻である。

【0153】

この問題への対応としては、命令単位の非プログラム順序実行と同様に、データ依存投機実行が有効である。すなわち、スレッドが含む命令間の正依存関係の

有無が判明するよりも以前に、正依存関係が存在しないと仮定して、投機的にスレッドの非プログラム順序実行を行う、スレッド単位のデータ依存投機実行である。

【 0 1 5 4 】

スレッド単位のデータ依存投機実行を実現するためには、順序関係が存在するスレッド対に関して、順序が前のスレッドが含むストア命令から、順序が後のスレッドが含むロード命令への正依存関係を検出する機能を必要とする。

【 0 1 5 5 】

本発明の第 2 の実施の形態におけるデータ依存関係検出装置は、このスレッド間の正依存関係を検出する機能を有する。ここで、本発明の第 2 の実施の形態におけるデータ依存関係検出装置は、スレッド間での非プログラム順序実行は行うが、スレッド内のスレッドを構成する命令間では非プログラム順序実行を行わないとして、スレッド内の命令間の正依存検出の機能は備えない。

【 0 1 5 6 】

次に図 6 を参照して、本発明の第 2 の実施の形態におけるデータ依存関係検出装置の構成を説明する。図 6 は、4 つのプロセッサ、すなわちプロセッサ 5 0、プロセッサ 5 1、プロセッサ 5 2、プロセッサ 5 3、及びスレッド制御部 5 5 から構成されるマルチプロセッサの構成を示すブロック図である。プロセッサ 5 0 ～ 5 3 はすべて同一の構成である。なお、図 6 において、プロセッサ 5 2 及びプロセッサ 5 3 の構成の詳細は省略しており、プロセッサ 5 2 及びプロセッサ 5 3 の同一の構成要素には同一の参照符号が付されている。

【 0 1 5 7 】

スレッド制御部 5 5 は、プロセッサ 5 0 ～ 5 3 の各々が実行するスレッドの割り当てや、スレッド間の順序関係の通知、スレッドのデータ依存投機実行が失敗した場合の回復処理の指示等、スレッド並列処理、及びデータ依存投機実行の制御を行う。プロセッサ 5 0 ～ 5 3 は、本発明の第 2 の実施の形態におけるデータ依存関係検出装置 3 0、プロセッサ制御部 5 7、及び命令実行部 1 8 を含む。プロセッサの他の構成要素については図示していない。

【 0 1 5 8 】

プログラムをプロセッサ 5 0 ~ 5 3 によってスレッド並列処理するにあたり、スレッド制御部 5 5 は、プロセッサ 5 0 ~ 5 3 の各々が実行すべきスレッドの割り当てを行う。また、プロセッサ 5 0 ~ 5 3 のそれぞれに、そのスレッド実行はデータ依存投機実行か否かを示す投機実行フラグや他の 3 つのプロセッサが実行するスレッドとの順序関係を示すスレッド順序を出力する。

【 0 1 5 9 】

プロセッサ 5 0 ~ 5 3 はそれぞれ、投機実行フラグ 3 9 及びスレッド順序 4 0 を入力する。

【 0 1 6 0 】

スレッド制御部 5 5 より割り当てられたスレッドを、各プロセッサが実行する場合、各プロセッサのプロセッサ制御部 5 7 は、実行する命令の種別を、各プロセッサの命令実行部 1 8 とデータ依存関係検出装置 3 0 に出力するとともに、実行命令 4 6 を通じて、他の 3 つのプロセッサに出力する。

【 0 1 6 1 】

データ依存関係検出装置 3 0 は、実行する命令種別を実行命令 3 4 から入力する。同時に、データ依存関係検出装置 3 0 は、他の 3 つのプロセッサが実行する命令種別を、実行命令 3 8 を通じてそれぞれ入力する。さらに、実行する命令がロード命令またはストア命令の場合、プロセッサ制御部 5 7 は、ロード命令またはストア命令の対象アドレスを、各プロセッサの命令実行部 1 8 とデータ依存関係検出装置 3 0 に出力するとともに、ロード・ストアアドレス 4 5 を通じて、他の 3 つのプロセッサに出力する。データ依存関係検出装置 3 0 は、ロード・ストア命令の対象アドレスをロード・ストアアドレス 3 5 から入力する。同時に、他の 3 つのプロセッサが実行する命令がロード命令またはストア命令の場合、データ依存関係検出装置 3 0 はその対象アドレスをロード・ストアアドレス 3 7 を通じてそれぞれ入力する。

【 0 1 6 2 】

例えば、プロセッサ 5 0 がスレッド制御部 5 5 より割り当てられたスレッドを実行する際、プロセッサ 5 0 のプロセッサ制御部 5 7 は実行する命令の種別をプロセッサ 5 0 の命令実行部 1 8 及びデータ依存関係検出装置 3 0 に加えて、実行

命令46を通じて他の3つのプロセッサ、プロセッサ51、プロセッサ52、プロセッサ53に出力する。同時に、プロセッサ50のデータ依存関係検出装置30は他の3つのプロセッサ、プロセッサ51、プロセッサ52、プロセッサ53が実行する命令種別を実行命令38を通じてそれぞれ入力する。

【0163】

さらに、実行する命令がロード命令またはストア命令の場合、プロセッサ50のプロセッサ制御部57は、その対象アドレスをプロセッサ50の命令実行部18及びデータ依存関係検出装置30、及びロード・ストアアドレス45を通じて他の3つのプロセッサ、プロセッサ51、プロセッサ52、プロセッサ53に出力する。同時に、他の3つのプロセッサ、プロセッサ51、プロセッサ52、プロセッサ53が実行する命令がロード命令またはストア命令の場合、プロセッサ50のデータ依存関係検出装置30はその対象アドレスをロード・ストアアドレス37を通じてそれぞれ入力する。

【0164】

プロセッサ50～53のそれぞれが備えるデータ依存関係検出装置30は、4つのハッシュ関数回路31、命令履歴テーブル32及び論理和回路41から構成される。また、自身が設置されているプロセッサ（「自プロセッサ」という）のプロセッサ制御部57からの入力として、実行命令34、ロード・ストアアドレス35、及び、他の3つのプロセッサから、それぞれの実行命令38とロード・ストアアドレス37を入力する。また、スレッド制御部55からの入力として、投機実行フラグ39及びスレッド順序40を備える。また、スレッド制御部55への出力として、データ依存検出結果36を備える。

【0165】

4つのハッシュ関数回路31は、ロード命令またはストア命令の対象アドレスを、命令履歴テーブル32のエントリ番号に変換するハッシュ関数 f を実現する論理回路である。ここで、上記ハッシュ関数 f は、同一の入力に対しては、同一の値をもつ。すなわち、 $N1 = f(A1)$ かつ $N2 = f(A2)$ としたとき、 $A1 = A2$ であれば $N1 = N2$ が成立する関数である。4つのハッシュ関数回路31のうちの1つは、自プロセッサが実行するロード命令の対象アドレスを命令履

歴テーブル32のエントリ番号に変換する。4つのハッシュ関数回路31のうちの残りの3つはそれぞれ、他の3つのプロセッサが実行するストア命令の対象アドレスを命令履歴テーブル32のエントリ番号に変換する。

【0166】

命令履歴テーブル32は、1ビットの情報を格納できる複数のエントリから構成される。命令履歴テーブル32が備えるエントリの総数は、ハッシュ関数回路31の出力がnビット幅である場合、2のn乗に等しい数である。命令履歴テーブル32は、1つの書き込みポートと3つの読み出しポートを備える。すなわち、1つの書き込み処理と、3つの読み出し処理を同時に行うことができる。

【0167】

4つのハッシュ関数回路31のうちの1つである、自プロセッサが実行するロード命令の対象アドレスを入力とするハッシュ関数回路31の出力は、命令履歴テーブル32の書き込みポートに接続され、ハッシュ関数回路31が出力するエントリ番号が指し示すエントリに対して書き込み処理を行う。

【0168】

4つのハッシュ関数回路31のうちの残りの3つである、他のプロセッサが実行するストア命令の対象アドレスを入力とするハッシュ関数回路31の出力は、命令履歴テーブル32の読み出しポートに接続され、ハッシュ関数回路31が出力するエントリ番号が指し示すエントリに対して読み出し処理を行う。

【0169】

さらに、スレッド制御部55からデータ依存関係検出装置30に入力される投機実行フラグ39に基づいて、すべてのエントリに同一の値を書き込む初期化機能を有する。

【0170】

論理和関数41は、命令履歴テーブル32が備える3つの読み出しポートの3つの読み出し結果に対して論理和を演算し、データ依存検出結果36としてスレッド制御部55に出力する。

【0171】

次に、図6を参照して、データ依存関係検出装置30の動作について説明する

。まず、初期状態においては、命令履歴テーブル 3 2 を構成するすべてのエントリは、自プロセッサがロード命令を実行していないことを示す第 1 の状態（例えば論理 0）にある。自プロセッサがスレッド間のデータ依存関係に対して投機的な命令実行を行わず、プログラム順序で確定的に命令実行を行う状態にある場合、データ依存関係検出装置 3 0 はなんら動作を行わない。

【 0 1 7 2 】

それに対して、自プロセッサがスレッド間のデータ依存関係に対して投機的な命令実行を行う状態にある場合、データ依存関係検出装置 3 0 は、他の 3 つのプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前の順序にあるスレッド（これを「先行するスレッド」という）を割り当てられたプロセッサが実行するストア命令から、自プロセッサが実行するロード命令への正依存関係の検出処理を行う。

【 0 1 7 3 】

自プロセッサが確定実行状態または投機実行状態のどちらにあるかは、スレッド制御部 5 5 からデータ依存関係検出装置 3 0 に入力される投機実行フラグ 3 9 で判断する。

【 0 1 7 4 】

また、自プロセッサが実行するスレッドが、他の 3 つのプロセッサが実行するそれぞれのスレッドよりもプログラム順序で前の順序にある否かは、スレッド制御部 5 5 からデータ依存関係検出装置 3 0 に入力されるスレッド順序 4 0 で判断する。また、他の 3 つのプロセッサが実行している命令の種別は、他の 3 つのプロセッサからデータ依存関係検出装置 3 0 に入力される実行命令 3 8 で判断する。

【 0 1 7 5 】

自プロセッサが投機実行状態にあるときに、投機的なロード命令が実行されると、データ依存関係検出装置 3 0 は、投機的なロード命令が実行されたことを命令履歴テーブル 3 2 に記憶する。より詳細には、ロード命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 3 2 のエントリ番号に変換し、命令履歴テーブル 3 2 の書き込みポートに入力する。命令履歴テーブル 3 2 は、入力

されたエントリ番号に対応するエントリを、投機的なロード命令が実行されたことを示す第 2 の状態（例えば論理 1）に変更する。

【 0 1 7 6 】

一方、自プロセッサが投機実行状態にあるときに、他の 3 つのプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前の順序にあるスレッド、すなわち先行するスレッドを割り当てられたプロセッサでストア命令が実行された場合には、データ依存関係検出装置 3 0 は、対応する命令履歴テーブル 3 2 のエントリに記憶された状態の読み出しを行う。より詳細には、ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 3 2 のエントリ番号に変換し、命令履歴テーブル 3 2 の 3 つの入力ポートのうち対応するものに入力する。命令履歴テーブル 3 2 は、入力されたエントリ番号に対応するエントリの内容を読み出し、論理和回路 4 1 に出力する。論理和回路 4 1 は、命令履歴テーブル 3 2 の 3 つの読み出しポートの出力の論理和を演算し、データ依存検出結果 3 6 としてスレッド制御部 5 5 に出力する。

【 0 1 7 7 】

他の 3 つのプロセッサのうち、自プロセッサが実行するスレッドよりも先行するスレッドを実行するプロセッサのいずれかが実行したストア命令の対象アドレスが、自プロセッサがそれまでデータ依存投機実行したロード命令の対象アドレスと等しいか、あるいは等しくない場合でも、エイリアスが発生して、命令履歴テーブル 3 2 の同じエントリに割り当てられた場合、命令履歴テーブル 3 2 からデータ依存投機実行されたロード命令が存在することを示す値が読み出されるため、正依存関係が存在する可能性があることが検出される。この場合、データ依存関係検出装置 3 0 は、データ依存投機実行が失敗したことを示す値（例えば論理 1）をデータ依存検出結果 3 6 を通じてスレッド制御部 5 5 に出力する。

【 0 1 7 8 】

スレッド制御部 5 5 は、プロセッサ 5 0 ～ 5 3 のいずれかより、データ依存投機実行の失敗の通知を受け取ると、その失敗通知を出力したプロセッサ及び、その失敗通知を出力したプロセッサが実行していたスレッドよりも、プログラム順序で後に位置するスレッドを実行しているプロセッサに対して、データ依存投機

実行の失敗による回復処理の実行要求を出力する。

【0179】

データ依存投機実行の失敗による回復処理の実行要求の出力対象となった各プロセッサでは、データ依存投機実行の失敗による回復処理の実行要求が、回復処理実行要求47を通じてプロセッサ制御部57に通知される。

【0180】

データ依存投機実行の失敗を通知したプロセッサだけではなく、データ依存投機実行を失敗したスレッドよりプログラム順序が後のスレッドを実行しているプロセッサに対しても、データ依存投機実行の失敗による回復処理の実行を要求するのは、次に示す理由による。すなわち、データ依存投機実行の失敗を通知をしたプロセッサは、正依存関係に違反してロード命令をデータ依存投機実行した可能性がある。そのため、前記データ依存投機実行の失敗を通知をしたプロセッサが実行していた、データ依存投機実行に失敗したスレッドの実行結果は正しくない可能性がある。従って、前記データ依存投機実行に失敗したスレッドの実行結果を参照している可能性のある、それよりもプログラム順序で後に位置するスレッドの実行結果もまた、正しくない可能性がある。

【0181】

同様にして、データ依存投機実行に失敗したスレッドの実行結果を参照している可能性のある前記スレッドの実行結果を参照している可能性のある、それよりもさらにプログラム順序で後に位置するスレッドの実行結果もまた、正しくない可能性がある。すなわち、データ依存投機実行の失敗を通知をしたプロセッサが実行していたスレッドよりもプログラム順序が後に位置するすべてのスレッドの実行結果は正しくない可能性がある。このため、データ依存投機実行の失敗を通知したプロセッサだけではなく、データ依存投機実行を失敗したスレッドよりプログラム順序が後のスレッドを実行しているプロセッサに対しても、データ依存投機実行の失敗による回復処理の実行を要求し、スレッドの実行結果の正しさを保証する。

【0182】

一方、データ依存投機実行の失敗により、スレッド制御部55からのデータ依

存投機実行の失敗による回復処理の実行要求の対象になったプロセッサでは、データ依存投機実行の失敗による回復処理の実行要求が、各プロセッサの回復処理実行要求 4 7 を通じてプロセッサ制御部 5 7 に通知される。その場合、上記各プロセッサは、データ依存投機実行を行ったスレッドの再実行を行うなど、データ依存投機実行の失敗による回復処理を実行することで、スレッドの実行結果の正しさを保証する。

【 0 1 8 3 】

上述したように、本発明の第 2 の実施の形態によるデータ依存関係検出装置 3 0 は、自プロセッサが実行するスレッドよりプログラム順序で先行するスレッドに含まれるストア命令から、自プロセッサが実行するスレッドに含まれるロード命令への正依存関係を検出する機能を有する。この機能は、データ依存関係検出装置 3 0 が備える命令履歴テーブル 3 2 が、自プロセッサがロード命令をデータ依存投機実行する際に書き込みを行う書き込みポートを備えると共に、他のプロセッサがストア命令を実行する際に読み出しを行う複数の読み出しポートを備えることで実現される。この機能によりスレッド並列処理において、スレッド単位のデータ依存投機実行の実現を可能とすることで、スレッド単位の並列実行によるプログラム実行のさらなる高速化を可能としている。

【 0 1 8 4 】

また、本発明の第 2 の実施の形態によるデータ依存関係検出装置 3 0 は、前述した本発明の第 1 の実施の形態によるデータ依存関係検出装置 1 0 と同様に、ストア命令からロード命令への正依存関係の存在を厳密に検出するのではなく、ストア命令からロード命令への正依存関係の存在の可能性を検出するものである。任意のロード命令の対象アドレスは、ハッシュ関数回路 3 1 により、命令履歴テーブル 3 2 の特定のエントリに重複を許して割り当てるため、従来のデータ依存関係検出装置と異なり、実行されたロード命令の対象アドレス自体をすべて記憶しておく必要がなく、またストア命令の対象アドレスと比較するためのアドレス比較器も必要としない。

【 0 1 8 5 】

すなわち、本発明の第 2 の実施の形態のデータ依存関係検出装置によれば、デ

ータ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。

【0186】

さらに、本発明の第2の実施の形態のデータ依存関係検出装置によれば、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもデータ依存投機実行による実行性能の向上を享受することのできる。特に、スレッド単位のデータ依存投機実行では、命令単位のデータ依存投機実行に比べてデータ依存投機実行されるロード・ストア命令数が多いため、データ依存投機実行することのできるロード命令の数がデータ依存関係検出装置のハードウェア量に制限されない、という本発明の特徴は、プログラム実行性能の向上に大きな効果を奏する。

【0187】

次に、図6及び図7を参照して、本発明の第2の実施の形態によるスレッド並列処理におけるデータ依存投機実行の具体例を以下に示す。まず、図7(a)は、ロード命令LD1、LD2、LD3、LD4、LD5、ストア命令ST1、ST2、ST3、及びロード・ストア命令以外の命令を含む12命令からなるプログラム例における命令のプログラム順序である。図中、空欄の四角はロード・ストア命令以外の命令を示す。

【0188】

プログラム順序は順に、ロード命令LD1、ロード・ストア命令以外の命令、ストア命令ST1、ロード命令LD2、ストア命令ST2、ロード・ストア命令以外の命令、ロード命令LD3、ロード・ストア命令以外の命令、ロード命令LD4、ロード命令LD5、ストア命令ST3、ロード・ストア命令以外の命令、の順序である。ここで、LD1の対象アドレスはA1、LD2の対象アドレスはA2、LD3の対象アドレスはA3、LD4の対象アドレスはA4、LD5の対象アドレスはA5、ST1の対象アドレスはA3、ST2の対象アドレスはA1、ST3の対象アドレスはA3、である。

【0189】

次に、図7(a)に示したプログラム順序のプログラム例を、図6に示した4

つのプロセッサ、プロセッサ50、プロセッサ51、プロセッサ52、プロセッサ53、においてスレッド単位でデータ依存投機実行を行うとする。具体的には、図7(a)に示した12命令から構成されるプログラム例を3命令から構成される4つのスレッドに分割し、それぞれを4つのプロセッサで並列実行を行う。すなわち、プログラム順序で順に、ロード命令LD1、ロード・ストア命令以外の命令、ストア命令ST1、の3命令列をスレッド0、ロード命令LD2、ストア命令ST2、ロード・ストア命令以外の命令、の3命令列をスレッド1、ロード命令LD3、ロード・ストア命令以外の命令、ロード命令LD4、の3命令列をスレッド2、ロード命令LD5、ストア命令ST3、ロード・ストア命令以外の命令、の3命令列をスレッド3として、スレッド0をプロセッサ52で、スレッド1をプロセッサ51で、スレッド2をプロセッサ50で、スレッド3をプロセッサ53で並列に実行するとする。この各プロセッサが実行するスレッドの割り当ては、スレッド制御部55によって行われる。

【0190】

LD3とST1の対象アドレスは同じA3であり、プログラム順序でST1の方が前にあるため、ST1からLD3にはデータ依存関係が存在する。すなわち、実行時においてLD3よりもST1が前に実行されて、ST1がアドレスA3に書き込みを行った結果を、LD3がアドレスA3から読み出しを行わなければ、プログラムの実行結果の正しさは保証されない。スレッド単位の並列処理においては、ST1を含むスレッド0からLD3を含むスレッド2へ正依存関係が存在することになる。しかしながら、通常、各プロセッサへのスレッドを割り当てる時点では、ST1からLD3への正依存関係、すなわちスレッド0からスレッド2への存在は判明しない。このため、確定的にスレッドの並列実行を行うとすると、正依存関係が無いことが判明していないスレッド間では、プログラム順序でスレッドを逐次に行うことで、正依存関係を保証する必要があるため、スレッドの並列処理によるプログラム実行の高速性を充分得ることができない。

【0191】

一方、スレッド単位のデータ依存投機実行では、スレッド間の正依存関係の存在の有無が判明していなくても、投機的にスレッドの並列実行を行う。データ依

存投機実行が成功する確率が充分高ければ、スレッドの並列処理によるプログラム実行の高速化が達成される。図7(b)は、図7(a)に示したプログラム順序の命令列、スレッド列のデータ依存投機実行によるスレッド並列実行の例であり、左から順にサイクル番号、スレッド2を実行するプロセッサ50のそのサイクルの実行命令、その実行命令の対象アドレス、スレッド1を実行するプロセッサ51のそのサイクルの実行命令、その実行命令の対象アドレス、スレッド0を実行するプロセッサ52のそのサイクルの実行命令、その実行命令の対象アドレス、スレッド3を実行するプロセッサ53のそのサイクルの実行命令、その実行命令の対象アドレス、をそれぞれ示す。すなわち、スレッド2を実行するプロセッサ50では、1サイクル目にLD3、2サイクル目にロード・ストア命令以外の命令、3サイクル目にLD4、が実行される。スレッド1を実行するプロセッサ51では、1サイクル目にLD2、2サイクル目にST2、3サイクル目にロード・ストア命令以外の命令、が実行される。スレッド0を実行するプロセッサ52では、1サイクル目にLD1、2サイクル目にロード・ストア命令以外の命令、3サイクル目にST1、が実行される。スレッド3を実行するプロセッサ53では、1サイクル目にLD5、2サイクル目にST3、3サイクル目にロード・ストア命令以外の命令、が実行される。

【0192】

ここで、スレッド0はプログラム順序で最も前に位置するスレッドのため、スレッド1～3に含まれるストア命令からスレッド0に含まれるロード命令への正依存関係は存在しない。すなわち、プロセッサ52は確定的にスレッド0を実行することができる。一方、スレッド1～3に関しては、正依存関係の有無が不明であるため、プロセッサ50、プロセッサ51、及びプロセッサ53はスレッド1～3をデータ依存投機実行することになる。図7(b)に示した実行例では、仮にデータ依存投機実行が成功した場合、12命令からなるプログラムが3サイクルで実行されることになり、大幅なプログラム実行性能の向上が達成される。

【0193】

しかしながら、図7(b)に示した実行例の場合、正依存関係のあるST1及びLD3について、LD3が1サイクル目にプロセッサ50で実行され、ST1

が3サイクル目にプロセッサ52で実行されるため、正依存関係が保証されておらず、正しいプログラムの実行結果が得られない。すなわち、プロセッサ50におけるスレッド2のデータ依存投機実行は失敗している。プロセッサ50が備えるデータ依存関係検出装置30は、プロセッサ52が実行したST1からプロセッサ50が実行したLD3への正依存関係の存在を検出し、スレッド2のデータ依存投機実行が失敗したことをスレッド制御部55に通知する機能を担う。スレッド制御部55は、プロセッサ50よりデータ依存投機実行が失敗した旨を通知されると、正しいプログラムの実行結果が得られるように、プロセッサ50及びデータ依存投機実行が失敗したスレッド2よりプログラム順序で後に位置するスレッド3を実行するプロセッサ53に対して、スレッドの実行結果の取り消しやスレッドの再実行など、データ依存投機実行失敗による回復処理を要求する。

【0194】

図7(c)は、図7(b)に示したプロセッサ50の命令実行順序における、プロセッサ50が備えるデータ依存関係検出装置30の動作を示す図である。図7(c)に左から順にサイクル番号、そのサイクルの命令履歴テーブル32の状態をそれぞれ示す。命令履歴テーブル32は4つのエントリから構成されるとき、4つのエントリの状態を右から順に、エントリ00、エントリ01、エントリ10、エントリ11、の内容を示している。

【0195】

0サイクル目では、命令履歴テーブル32は初期状態にあり、すべてのエントリの内容はロード命令が実行されていないことを示す論理0である。次に、1サイクル目ではプロセッサ50において、スレッド2に含まれるロード命令LD3がデータ依存投機実行される。LD3の対象アドレスA3がハッシュ関数回路31により、命令履歴テーブル32のエントリ番号00に変換されたとすると、命令履歴テーブル32のエントリ00の内容をロード命令が実行されたことを示す論理1にする。

【0196】

次に、2サイクル目ではプロセッサ51においてスレッド1に含まれるST2が、プロセッサ53においてスレッド3に含まれるST3が実行される。プロセ

ッサ51が実行しているスレッド1は、プロセッサ50がデータ依存投機実行しているスレッド2よりもプログラム順序で前に位置するため、正依存検出の対象となる。他のプロセッサが実行している命令種別は実行命令38で、その実行命令の対象アドレスはロード・ストアアドレス37で、データ依存関係検出装置30に与えられている。ST2の対象アドレスA2がハッシュ関数回路31により命令履歴テーブル32のエントリ番号10に変換されるとすると、データ依存関係検出装置30は命令履歴テーブル32のエントリ10の内容を読み出して、ST2から自プロセッサ（プロセッサ50）が実行したロード命令への正依存関係の存在の有無を判定する。この場合、命令履歴テーブル32のエントリ10からは、ロード命令が実行されていないことを示す論理0が読み出されるため、プロセッサ50におけるスレッド2のデータ依存投機実行は、プロセッサ51が実行しているスレッド1に対して成功していると判断される。

【0197】

一方、プロセッサ53が実行しているスレッド3は、プロセッサ50がデータ依存投機実行しているスレッド2よりもプログラム順序で後に位置するため、正依存検出の対象とならない。すなわち、スレッド3に含まれるST3の対象アドレスA3は、プロセッサ50が1サイクル目に実行したLD3の対象アドレスと同じであるが、プログラム順序はLD3が前、ST3が後であるため、正依存関係は常に存在しない。このため、プロセッサ50が備える命令履歴テーブル32は、プロセッサ53が実行するスレッド3に含まれるストア命令に対しては、データ依存投機実行の成否判定を行わない。結局、2サイクル目において、プロセッサ50が備える命令履歴テーブル32は、データ依存投機実行が成功したとして、そのままスレッド実行を続行する。

【0198】

次に、3サイクル目ではプロセッサ50においてスレッド2に含まれるLD4が、プロセッサ52においてスレッド0に含まれるST1が実行される。まず、LD4に対してプロセッサ50が備える命令履歴テーブル32は、LD4の対象アドレスA4がハッシュ関数回路31により、命令履歴テーブル32のエントリ番号11に変換されるとすると、命令履歴テーブル32のエントリ1の内容をロ

ード命令が実行されたことを示す論理1にする。一方、プロセッサ52が実行しているスレッド0は、プロセッサ50がデータ依存投機実行しているスレッド2よりもプログラム順序で前に位置し、正依存検出の対象となるため、プロセッサ52が実行するST1に対しては、自プロセッサ（プロセッサ50）が実行したロード命令への正依存関係の存在の有無を判定する。具体的には、ST1の対象アドレスA3はハッシュ関数回路31により命令履歴テーブル32のエントリ番号00に変換されるので、データ依存関係検出装置30は命令履歴テーブル32のエントリ00の内容を読み出して、ST1から自プロセッサ（プロセッサ50）が実行したロード命令への正依存関係の存在の有無を判定する。この場合、命令履歴テーブル32のエントリ00からは、0サイクル目にLD3により格納された論理1を読み出される。すなわち、ST1からLD3への正依存関係が検出されるため、プロセッサ50におけるスレッド2のデータ依存投機実行は、プロセッサ52が実行しているスレッド0に対して失敗したと判断され、スレッド制御部55にその旨がデータ依存検出結果36を通じて通知される。

【0199】

スレッド制御部55は、プロセッサ50よりデータ依存投機実行が失敗した旨を通知されると、正しいプログラムの実行結果が得られるように、プロセッサ50及びデータ依存投機実行が失敗したスレッド2よりプログラム順序で後に位置するスレッド3を実行するプロセッサ53に対して、スレッドの実行結果の取り消し及び、スレッド再実行を要求し、データ依存投機実行失敗の回復を図る。

【0200】

以上、プロセッサ50について、データ依存投機実行における、データ依存関係検出装置30による正依存関係の検出及びデータ依存投機実行の正否判定の動作について説明したが、スレッド1を実行するプロセッサ51、スレッド0を実行するプロセッサ52、及びスレッド3を実行するプロセッサ53も同様に、それぞれが備えるデータ依存関係検出装置30により、正依存関係の検出及びデータ依存投機実行の正否判定が行われる。プロセッサ51及びプロセッサ52においては、正依存関係が検出されず、データ依存投機実行が成功したと判断されるため、3サイクル目で実行が完了する。一方、プロセッサ53においても正依存

関係が検出されず、データ依存投機実行が成功したと判断されるが、プロセッサ 53 が実行するスレッド 3 よりもプログラム順序で前に位置するスレッド 2 を実行するプロセッサ 50 がデータ依存投機実行を失敗したため、データ依存投機実行失敗による回復処理が要求される。

【 0 2 0 1 】

スレッド制御部 55 よりデータ依存投機実行失敗による回復処理が要求されるプロセッサ 50 及びプロセッサ 53 は、回復処理実行要求 47 を通じてそれぞれのプロセッサ制御部 57 がその要求を受け取ると、まず 3 サイクル目までのスレッドの実行結果の取り消しを行う。次に、5 サイクル目からスレッドの再実行を開始する。スレッド 0 及びスレッド 1 が既に実行を完了しているため、スレッド 2 は実行が完了していないスレッド、すなわちスレッド 2 及びスレッド 3 のうち、プログラム順序で一番前にあるスレッドであり、従って確定的に実行することができる。具体的には、プロセッサ 50 は、5 サイクル目に LD 3、6 サイクル目にロード・ストア命令以外の命令、7 サイクル目に LD 4、を確定的に再実行する。5 サイクル目に実行する LD 3 に対しては、スレッド 0 に含まれる ST 1 から正依存関係が存在するが、ST 1 は既にプロセッサ 52 により 3 サイクル目に実行されているため、正依存関係に違反しない。一方、スレッド 3 は、スレッド 2 が含むストア命令から、スレッド 3 が含むロード命令への正依存関係、すなわち、スレッド 2 からの正依存関係の存在が不明であるため、再びデータ依存投機実行されることになる。具体的には、プロセッサ 53 は、5 サイクル目に LD 5、6 サイクル目に ST 3、7 サイクル目にロード・ストア命令以外の命令、を投機的に再実行する。この場合、スレッド 2 からの正依存関係は検出されないため、データ依存投機実行は成功したとして、7 サイクル目でスレッド 3 の実行は完了する。

【 0 2 0 2 】

以上、本発明の第 2 の実施の形態によるデータ依存投機実行の具体例を説明した。12 命令からなるプログラム例に対して、4 つのプロセッサによりスレッド並列処理を行うと、データ依存投機実行が成功した場合は 3 サイクルで、データ依存投機実行が 1 回失敗した場合でも 7 サイクルで実行が完了し、プログラムの

実行性能の大幅は向上を望むことが可能となる。上記に示した具体例では、説明のため各スレッドを3命令から構成したが、通常は数十命令から数万命令で構成されるので、より多くのロード命令がデータ依存投機実行されることになる。しかしながら、ロード命令の対象アドレスをすべて記憶する必要のある従来のデータ依存関係検出装置100では、アドレスバッファ101に空きがなくなるとデータ依存投機実行することができないため、それ以降は逐次的かつ確定的にスレッド実行をせざるおえない。すなわち、従来のデータ依存関係検出装置100では、スレッド並列処理におけるデータ依存投機実行によるプログラム実行の高速化の効果を十分に享受することができない、あるいは、享受するためには膨大なハードウェア量が必要となるという問題がある。

【0203】

一方、本発明の第2の実施の形態によるデータ依存関係検出装置30は、ロードストア命令の対象アドレスに関してエイリアスの存在を許容することで、ロード命令の対象アドレスそのものをすべて記憶する必要をなくしたことを特徴としている。この特徴により、データ依存投機実行が可能なロード命令数に制限がなく、必要なハードウェア量も小さい、という効果を得ることができる。また、正依存関係の検出の処理時間が短いため、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。さらに、データ依存関係検出装置30は、スレッド制御部55が出力するスレッド順序40に基づいて、正依存検出する対象のスレッドを選択する機能を有することを特徴としている。この特徴により、スレッド間の順序に柔軟に対応して、スレッド間の正依存関係を適切に検出できる、という効果を得ることができる。

【0204】

上記したデータ依存関係検出装置30では、スレッド制御部55が出力するスレッド順序40を参照して、自プロセッサよりも先行するスレッドを実行している他のプロセッサを特定し、それら先行するスレッドを実行するプロセッサがストア命令を実行するときのみ、命令履歴テーブル32の読み出しを行い、次に論理和関数41でそれら読み出し結果の論理和を演算することで、先行するスレッドから自プロセッサが実行するスレッドへの正依存関係の存在の検出を実現した

。一方、他の実現方法として、自プロセッサよりも先行するしかないにもかかわらず、他のプロセッサがストア命令を実行した場合には、常に命令履歴テーブル32の読み出しを行っても良い。この場合、論理和関数41でそれら読み出し結果の論理和を演算する際に、スレッド制御部55が出力するスレッド順序40を参照して、自プロセッサよりも先行するスレッドを実行している他のプロセッサを特定し、それら先行するスレッドによる読み出した結果のみ論理和を演算することで、先行するスレッドから自プロセッサが実行するスレッドへの正依存関係の存在の検出を実現することができる。

【0205】

また、上記した本発明の第2の実施の形態は、スレッド並列処理におけるスレッド単位のデータ依存投機実行において、自プロセッサでロード命令が実行された場合には、該ロード命令が処理対象とするアドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリに、ロード命令が実行されたことを示す値を格納し、他のプロセッサでストア命令が実行された場合には、該ストア命令が処理対象とするアドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリから、自プロセッサでロード命令が実行されたか否かを示す値を読み出すことで、他のプロセッサで実行されたストア命令から、自プロセッサで実行されたロード命令への正依存関係が存在する可能性を検出することを特徴とした。しかしながら、この実施の形態は、スレッド間の正依存関係が存在する可能性の検出にとどまらず、スレッド間で逆依存関係や出力依存関係が存在する可能性の検出に対しても、同様の構成で容易に適用することができる。

【0206】

例えば、自プロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリに、自プロセッサでストア命令が実行されたことを示す値を格納し、他のプロセッサでロード命令が実行された場合には、該ロード命令の対象アドレスをハ

ッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出すことで、他のプロセッサで実行されたロード命令から、自プロセッサで実行されたストア命令への逆依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置が実現される。

【0207】

または、他のプロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、自プロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリに、自プロセッサでストア命令が実行されたことを示す値を格納することで、他のプロセッサで実行されたストア命令から、自プロセッサで実行されたストア命令への出力依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置が実現される。

【0208】

さらに、他のプロセッサでロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、他のプロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル32のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル32のエントリから、自プロセッサでロード命令が実行されたか否かを示す値及び自プロセッサでストア命令が実行されたか否かを示す値を読み出し、自プロセッサでロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路31に

より命令履歴テーブル 3 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 3 2 のエントリに、自プロセッサでロード命令が実行されたことを示す値を格納し、自プロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 3 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 3 2 のエントリに、自プロセッサでストア命令が実行されたことを示す値を格納することで、他プロセッサで実行されたストア命令から、自プロセッサで実行されたロード命令への正依存関係が存在する可能性、及び他プロセッサで実行されたロード命令から、自プロセッサで実行されたストア命令への逆依存関係が存在する可能性、及び他のプロセッサで実行されたストア命令から、自プロセッサで実行されたストア命令への出力依存関係が存在する可能性、のすべてを検出する機能を有するデータ依存関係検出装置が実現される。

【 0 2 0 9 】

また、上記した本発明の第 2 の実施の形態によるデータ依存関係検出装置 3 0 は、スレッド間の正依存関係を検出する機能は有するが、スレッド内の命令間の正依存関係を検出する機能は有しない。すなわち、スレッド並列処理におけるスレッド単位のデータ依存投機実行は可能であるが、スレッド内の命令単位のデータ依存投機実行は行うことができない。そこで、スレッド間の正依存関係の検出機能のみならず、スレッド内の命令間の正依存関係の検出機能も備えることで、スレッド単位のデータ依存投機実行を実現するのに加えて、命令単位のデータ依存投機実行を可能とした、データ依存関係検出装置の第 3 の実施の形態を次に示す。

【 0 2 1 0 】

図 8 を参照して、本発明の第 3 の実施の形態におけるデータ依存関係検出装置 6 0 の構成を説明する。図 8 は、本発明の第 3 の実施の形態におけるデータ依存関係検出装置 6 0 を備え、プロセッサ制御部 6 3、及び命令実行部 1 8 を含むプロセッサ 5 9 の構成を示すブロック図である。プロセッサの他の構成要素や、他のプロセッサ、スレッド制御部などは図示していない。プロセッサ 5 9 の大部分は、本発明の第 2 の実施の形態によるデータ依存関係検出装置 3 0 を備えるプロ

セッサ 5 0 ～ 5 3 と同様である。

【 0 2 1 1 】

スレッド制御部（図示せず）より割り当てられたスレッドを、プロセッサ 5 9 が実行する場合、プロセッサ 5 9 が備えるプロセッサ制御部 6 3 は、スレッドを構成する命令を、プログラム順序で確定的に実行するか、非プログラム順序でデータ依存投機実行を行うか、を示す投機実行フラグ 3 3 をデータ依存関係検出装置 6 0 に出力する。また、プロセッサ制御部 6 3 は、実行する命令の種別を命令実行部 1 8 及びデータ依存関係検出装置 6 0、及び実行命令 4 6 を通じて他の 3 つのプロセッサに出力する。データ依存関係検出装置 6 0 はその実行する命令種別を実行命令 3 4 から入力する。同時に、データ依存関係検出装置 6 0 は他の 3 つのプロセッサが実行する命令種別を実行命令 3 8 を通じてそれぞれ入力する。さらに、実行する命令がロード命令またはストア命令の場合、プロセッサ制御部 6 3 は、その対象アドレスを命令実行部 1 8 及びデータ依存関係検出装置 6 0、及びロード・ストアアドレス 4 5 を通じて他の 3 つのプロセッサに出力する。データ依存関係検出装置 6 0 はそのロード・ストア命令の対象アドレスをロード・ストアアドレス 3 5 から入力する。同時に、他の 3 つのプロセッサが実行する命令がロード命令またはストア命令の場合、データ依存関係検出装置 6 0 はその対象アドレスをロード・ストアアドレス 3 7 を通じてそれぞれ入力する。

【 0 2 1 2 】

データ依存関係検出装置 6 0 は、4 つのハッシュ関数回路 3 1、命令履歴テーブル 6 2 及び論理和回路 6 1 から構成される。また、自プロセッサのプロセッサ制御部 6 3 からの入力として、投機実行フラグ 3 3、実行命令 3 4、ロード・ストアアドレス 3 5 及び、他の 3 つのプロセッサから、それぞれの実行命令 3 8 とロード・ストアアドレス 3 7 を入力する。また、スレッド制御部からの入力として、投機実行フラグ 3 9 及びスレッド順序 4 0 を備える。また、スレッド制御部への出力として、データ依存検出結果 6 4 を備える。

【 0 2 1 3 】

4 つのハッシュ関数回路 3 1 は、ロード命令またはストア命令の対象アドレスを、命令履歴テーブル 6 2 のエントリ番号に変換するハッシュ関数 f を実現する

論理回路である。ここで、上記ハッシュ関数 f は、同一の入力に対しては、同一の値をもつことを特徴とする。すなわち、 $N1 = f(A1)$ かつ $N2 = f(A2)$ としたとき、 $A1 = A2$ であれば $N1 = N2$ が成立する関数である。4つのハッシュ関数回路31のうちの1つは、自プロセッサが実行するロード命令及びストア命令の対象アドレスを命令履歴テーブル62のエントリ番号に変換する。4つのハッシュ関数回路31のうちの残りの3つはそれぞれ、他の3つのプロセッサが実行するストア命令の対象アドレスを命令履歴テーブル62のエントリ番号に変換する。

【0214】

命令履歴テーブル62は、1ビットの情報を格納できる複数のエントリから構成される。備える上記エントリの総数は、ハッシュ関数回路31の出力が n ビット幅である場合、2の n 乗に等しい数である。命令履歴テーブル62は、1つの読み出し書き込みポートと3つの読み出しポートを備えることを特徴とする。すなわち、1つの読み出し処理または書き込み処理と、3つの読み出し処理を同時に行うことができる。4つのハッシュ関数回路31のうちの1つである、自プロセッサが実行するロード命令またはストア命令の対象アドレスを入力とするハッシュ関数回路31の出力は、命令履歴テーブル62の読み出し書き込みポートに接続され、ハッシュ関数回路31が出力するエントリ番号が指し示すエントリに対して読み出し処理または書き込み処理を行う。4つのハッシュ関数回路31のうちの残りの3つである、他のプロセッサが実行するストア命令の対象アドレスを入力とするハッシュ関数回路31の出力は、命令履歴テーブル62の読み出しポートに接続され、ハッシュ関数回路31が出力するエントリ番号が指し示すエントリに対して読み出し処理を行う。さらに、スレッド制御部からデータ依存関係検出装置60に入力される投機実行フラグ39及び、プロセッサ制御部63からデータ依存関係検出装置60に入力される投機実行フラグ33に基づいて、すべてのエントリに同一の値を書き込む初期化機能を有する。論理和関数41は、命令履歴テーブル62が備える1つの読み出し書き込みポート及び3つの読み出しポートの合計4つの読み出し結果に対して論理和を演算し、データ依存検出結果64としてスレッド制御部に出力する。

【 0 2 1 5 】

次に、上記構成のデータ依存関係検出装置 6 0 の動作を説明する。まず、初期状態においては、命令履歴テーブル 6 2 を構成するすべてのエントリは、自プロセッサがロード命令を実行していないことを示す第 1 の状態（例えば論理 0）にある。自プロセッサがスレッド間のデータ依存関係に対して投機的なスレッド実行を行わず、プログラム順序で確定的にスレッド実行を行う状態にある場合であり、かつ自プロセッサがスレッド内の命令間のデータ依存関係に対して投機的な命令実行を行わず、プログラム順序で確定的に命令実行を行う状態にある場合、データ依存関係検出装置 6 0 はなんら動作を行わない。

【 0 2 1 6 】

それに対して、自プロセッサがスレッド間のデータ依存関係に対して投機的なスレッド実行を行う状態にある場合、データ依存関係検出装置 6 0 は、他の 3 つのプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前の順序にあるスレッドを割り当てられたプロセッサが実行するストア命令から、自プロセッサが実行するロード命令への正依存関係の検出処理を行う。自プロセッサがスレッド実行に関して確定実行状態または投機実行状態のどちらにあるかは、スレッド制御部からデータ依存関係検出装置 6 0 に入力される投機実行フラグ 3 9 で判断する。また、自プロセッサが実行するスレッドが、他の 3 つのプロセッサが実行するそれぞれのスレッドよりもプログラム順序で前の順序にある否かは、スレッド制御部からデータ依存関係検出装置 6 0 に入力されるスレッド順序 4 0 で判断する。また、他の 3 つのプロセッサが実行している命令の種別は、他の 3 つのプロセッサからデータ依存関係検出装置 6 0 に入力される実行命令 3 8 で判断する。

【 0 2 1 7 】

また、自プロセッサがスレッド内の命令間のデータ依存関係に対して投機的な命令実行を行う状態にある場合、データ依存関係検出装置 6 0 は、自プロセッサがそれ以前に実行したストア命令から、自プロセッサが実行するロード命令への正依存関係の検出処理を行う。自プロセッサがスレッド内の命令の実行に関して確定実行状態または投機実行状態のどちらにあるかは、プロセッサ制御部 6 3 か

らデータ依存関係検出装置 6 0 に入力される投機実行フラグ 3 3 で判断する。自プロセッサが実行する命令の種別は、プロセッサ制御部 6 3 からデータ依存関係検出装置 6 0 に入力される実行命令 3 4 で判断する。

【 0 2 1 8 】

自プロセッサがスレッド実行に関して、あるいはスレッド内の命令実行に関して投機実行状態にあるときに、投機的なロード命令が実行されると、データ依存関係検出装置 6 0 は、投機的なロード命令が実行されたことを命令履歴テーブル 6 2 に記憶する。より詳細には、ロード命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、命令履歴テーブル 6 2 の読み出し書き込みポートに入力する。命令履歴テーブル 6 2 は、入力されたエントリ番号に対応するエントリを、投機的なロード命令が実行されたことを示す第 2 の状態（例えば論理 1）に変更する。

【 0 2 1 9 】

一方、自プロセッサがスレッド実行に関して投機実行状態にあるときに、他の 3 つのプロセッサのうち、自プロセッサが実行するスレッドよりもプログラム順序で前の順序にあるスレッド、すなわち先行するスレッドを割り当てられたプロセッサでストア命令が実行された場合には、データ依存関係検出装置 6 0 の対応するエントリに記憶された状態の読み出しを行う。より詳細には、ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、命令履歴テーブル 6 2 の 3 つの入力ポートのうち対応するものに入力する。命令履歴テーブル 6 2 は、入力されたエントリ番号に対応するエントリの内容を読み出し、論理和回路 6 1 に出力する。同時に、自プロセッサがスレッド内の命令実行に関して投機実行状態にあるときに、自プロセッサがストア命令を実行した場合には、データ依存関係検出装置 6 0 の対応するエントリに記憶された状態の読み出しを行う。より詳細には、ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、命令履歴テーブル 6 2 の読み出し書き込み力ポートに入力する。命令履歴テーブル 6 2 は、入力されたエントリ番号に対応するエントリの内容を読み出し、論理和回路 6 1 に出力する。論理和回路 6 1 は、命令履歴テーブル 6 2 の 1 つの読み出し書き

込みポート及び3つの読み出しポートの出力の論理和を演算し、データ依存検出結果64としてスレッド制御部に出力する。

【0220】

スレッド実行をデータ依存投機実行していた場合、他の3つのプロセッサのうち、自プロセッサが実行するスレッドよりも先行するスレッドを実行するプロセッサのいずれかが実行したストア命令の対象アドレスが、自プロセッサがそれまでデータ依存投機実行したロード命令の対象アドレスと等しいか、あるいは等しくない場合でも、エイリアスが発生して、命令履歴テーブル62の同じエントリに割り当てられた場合、命令履歴テーブル62からデータ依存投機実行されたロード命令が存在することを示す値が読み出されるため、スレッド間で正依存関係が存在する可能性があることが検出される。この場合、データ依存関係検出装置60は、データ依存投機実行が失敗したことを示す値（例えば論理1）をデータ依存検出結果64を通じてスレッド制御部に出力する。

【0221】

さらに、スレッド内の命令実行をデータ依存投機実行していた場合、自プロセッサが実行したストア命令の対象アドレスが、自プロセッサがそれまでデータ依存投機実行したロード命令の対象アドレスと等しいか、あるいは等しくない場合でも、エイリアスが発生して、命令履歴テーブル62の同じエントリに割り当てられた場合、命令履歴テーブル62からデータ依存投機実行されたロード命令が存在することを示す値が読み出されるため、スレッド内の命令間で正依存関係が存在する可能性があることが検出される。この場合も、データ依存関係検出装置60は、データ依存投機実行が失敗したことを示す値（例えば論理1）をデータ依存検出結果64を通じてスレッド制御部に出力する。

【0222】

スレッド制御部は、プロセッサ59を含むデータ依存投機実行を行っていたプロセッサより、データ依存投機実行の失敗の通知を受け取ると、その失敗通知を出力したプロセッサ及び、その失敗通知を出力したプロセッサが実行していたスレッドよりも、プログラム順序で後に位置するスレッドを実行しているプロセッサに対して、データ依存投機実行の失敗による回復処理の実行要求を出力する。

データ依存投機実行の失敗による回復処理の実行要求の出力対象となった各プロセッサでは、データ依存投機実行の失敗による回復処理の実行要求が、回復処理実行要求 4 7 を通じてプロセッサ制御部 6 3 に通知される。

【 0 2 2 3 】

スレッド単位のデータ依存投機実行の失敗だけではなく、スレッド内の命令単位のデータ依存投機実行の失敗によっても、データ依存投機実行を失敗したスレッドよりプログラム順序が後のスレッドを実行しているプロセッサに対して、データ依存投機実行の失敗による回復処理の実行を要求するのは、次に示す理由による。すなわち、スレッド内の命令のデータ依存投機実行が失敗したプロセッサは、正依存関係に違反してロード命令をデータ依存投機実行した可能性がある。そのため、前記データ依存投機実行が失敗したプロセッサが実行していた、データ依存投機実行に失敗したスレッドの実行結果は正しくない可能性がある。従って、前記データ依存投機実行に失敗したスレッドの実行結果を参照している可能性のある、失敗したスレッドよりも後続のスレッドを実行しているプロセッサに対しても、データ依存投機実行の失敗による回復処理が必要となる。

【 0 2 2 4 】

スレッド単位あるいはスレッド内の命令単位のデータ依存投機実行の失敗により、データ依存投機実行の失敗による回復処理の実行要求の対象になったプロセッサでは、データ依存投機実行の失敗による回復処理の実行要求が、各プロセッサの回復処理実行要求 4 7 を通じてプロセッサ制御部 6 3 に通知される。その場合、上記各プロセッサは、データ依存投機実行を行ったスレッドの再実行を行うなど、データ依存投機実行の失敗による回復処理を実行することで、スレッドの実行結果の正しさを保証する。

【 0 2 2 5 】

上述したように、この実施の形態によるデータ依存関係検出装置 6 0 は、自プロセッサが実行するスレッドよりプログラム順序で先行するスレッドに含まれるストア命令から、自プロセッサが実行するスレッドに含まれるロード命令への正依存関係を検出する機能に加えて、自プロセッサが実行するスレッド内の命令間の正依存関係、すなわち、スレッド内のストア命令からロード命令への正依存関

係を検出する機能を有する。この2つの機能は、データ依存関係検出装置60が備える命令履歴テーブル62が、自プロセッサがロード命令をデータ依存投機実行する際に書き込みを行い、自プロセッサがストア命令をデータ依存投機実行する際に読み出しを行う読み出し書き込みポートを備えると共に、他のプロセッサがストア命令を実行する際に読み出しを行う複数の読み出しポートを備えることで実現される。これら機能により、スレッド並列処理において、スレッド単位のデータ依存投機実行の実現を可能とすることで、スレッド単位の並列実行によるプログラム実行のさらなる高速化を可能とするのみならず、スレッド内を構成する命令単位のデータ依存投機実行の実現を可能とすることで、スレッド内の非プログラム順序実行による高速化をも可能とする、という大きな効果を得ることができる。

【0226】

上記したデータ依存関係検出装置60では、スレッド制御部が出力するスレッド順序40を参照して、自プロセッサよりも先行するスレッドを実行している他のプロセッサを特定し、それら先行するスレッドを実行するプロセッサがストア命令を実行するときのみ、命令履歴テーブル62の読み出しを行い、次に論理和関数61でそれら読み出し結果の論理和を演算することで、先行するスレッドから自プロセッサが実行するスレッドへの正依存関係の存在の検出を実現した。一方、他の実現方法として、自プロセッサよりも先行するしないにかかわらず、他のプロセッサがストア命令を実行した場合には、常に命令履歴テーブル62の読み出しを行っても良い。この場合、論理和関数61でそれら読み出し結果の論理和を演算する際に、スレッド制御部が出力するスレッド順序40を参照して、自プロセッサよりも先行するスレッドを実行している他のプロセッサを特定し、それら先行するスレッドによる読み出した結果のみ論理和を演算することで、先行するスレッドから自プロセッサが実行するスレッドへの正依存関係の存在の検出を実現することができる。

【0227】

また、上記した本発明の第3の実施の形態は、スレッド並列処理におけるスレッド単位のデータ依存投機実行において、自プロセッサでロード命令が実行され

た場合には、該ロード命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル62のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル62のエントリに、自プロセッサでロード命令が実行されたことを示す値を格納し、自プロセッサあるいは他のプロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル62のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル62のエントリから、自プロセッサでロード命令が実行されたか否かを示す値を読み出すことで、自プロセッサあるいは他のプロセッサで実行されたストア命令から、自プロセッサで実行されたロード命令への正依存関係が存在する可能性を検出することを特徴とした。しかしながら、この実施の形態は、スレッド間及びスレッド内の命令間の正依存関係が存在する可能性の検出にとどまらず、スレッド間及びスレッド内の命令間の逆依存関係や出力依存関係が存在する可能性の検出に対しても、同様の構成で容易に適用することができる。

【0228】

例えば、自プロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル62のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル62のエントリに、自プロセッサでストア命令が実行されたことを示す値を格納し、自プロセッサあるいは他のプロセッサでロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル62のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル62のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出すことで、自プロセッサあるいは他のプロセッサで実行されたロード命令から、自プロセッサで実行されたストア命令への逆依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置が実現される。

【0229】

または、他のプロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路31により命令履歴テーブル62のエントリ番

号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、自プロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出すのに加えて、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリに、自プロセッサでストア命令が実行されたことを示す値を格納することで、自プロセッサあるいは他のプロセッサで実行されたストア命令から、自プロセッサで実行されたストア命令への出力依存関係が存在する可能性を検出する機能を有するデータ依存関係検出装置が実現される。

【 0 2 3 0 】

さらに、自プロセッサあるいは他のプロセッサでロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリから、自プロセッサでストア命令が実行されたか否かを示す値を読み出し、自プロセッサあるいは他のプロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリから、自プロセッサでロード命令が実行されたか否かを示す値及び自プロセッサでストア命令が実行されたか否かを示す値を読み出し、また、自プロセッサでロード命令が実行された場合には、該ロード命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリに、自プロセッサでロード命令が実行されたことを示す値を格納し、自プロセッサでストア命令が実行された場合には、該ストア命令の対象アドレスをハッシュ関数回路 3 1 により命令履歴テーブル 6 2 のエントリ番号に変換し、前記変換されたエントリ番号に対応する命令履歴テーブル 6 2 のエントリに、自プロセッサでストア命令が実行されたことを示す値を格納することで、自

プロセッサあるいは他プロセッサで実行されたストア命令から、自プロセッサで実行されたロード命令への正依存関係が存在する可能性、及び自プロセッサあるいは他プロセッサで実行されたロード命令から、自プロセッサで実行されたストア命令への逆依存関係が存在する可能性、及び自プロセッサあるいは他のプロセッサで実行されたストア命令から、自プロセッサで実行されたストア命令への出力依存関係が存在する可能性、のすべてを検出する機能を有するデータ依存関係検出装置が実現される。

【 0 2 3 1 】

上記した各実施例において、データ依存関係検出装置を備えたプロセッサは、好ましくは半導体基板上に集積化される半導体集積回路（L S I）装置として実現される。

【 0 2 3 2 】

【発明の効果】

以上説明したように、本発明のデータ依存関係検出装置によれば、データ依存投機実行時において、ストア命令からロード命令への正依存関係の存在を厳密に検出するのではなく、ストア命令からロード命令への正依存関係の存在の可能性を検出することを特徴としている。任意のロード命令の対象アドレスは、ハッシュ関数回路により、命令履歴テーブルの特定のエントリに重複を許して割り当てるため、従来のデータ依存関係検出装置と異なり、実行されたロード命令の対象アドレス自体をすべて記憶しておく必要がなく、またストア命令の対象アドレスと比較するためのアドレス比較器も必要としない。すなわち、本発明によるデータ依存関係検出装置によれば、データ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。また、本発明のデータ依存関係検出装置によれば、ストア命令の実行時の正依存関係の検出処理において、ストア命令の対象アドレスとロード命令の対象アドレスの比較を必要としない。さらに、命令履歴テーブルの読み出すエントリは1つだけなので、

すべてのエントリ出力の論理和を演算する必要もない。すなわち、正依存関係の検出の処理時間が短いため、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【 0 2 3 3 】

また、本発明のデータ依存関係検出装置によれば、スレッド並列処理におけるスレッド単位のデータ依存投機実行においても、データ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、データ依存投機実行を行うことのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもスレッド単位のデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。さらに、スレッド間の依存関係の検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【 0 2 3 4 】

さらに、本発明のデータ依存関係検出装置によれば、スレッド並列処理におけるスレッド単位のデータ依存投機実行及び、スレッド内の命令単位のデータ依存投機実行を可能とし、さらにそれらデータ依存投機実行の成否判定に必要なハードウェア量が小さいという効果を得ることができる。さらに、データ依存投機実行することのできるロード命令の数が、データ依存関係検出装置のハードウェア量に制限されないため、少ないハードウェア量でもスレッド単位のデータ依存投機実行及びスレッド内の命令単位のデータ依存投機実行による実行性能の向上を享受することのできるという効果を得ることができる。さらに、データ依存関係の検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易であるという効果を得ることができる。

【図面の簡単な説明】

【図 1】

本発明の第 1 の実施の形態によるデータ依存関係検出装置を含むプロセッサの構成を示すブロック図である。

【図 2】

本発明の第 1 の実施の形態によるデータ依存関係検出装置の動作を示すタイミング図である。

【図 3】

本発明の第 1 の実施の形態によるデータ依存関係検出装置の構成を示すブロック図である。

【図 4】

本発明の第 1 の実施の形態によるデータ依存関係検出装置の動作の具体例を説明する説明図である。

【図 5】

本発明のデータ依存関係検出装置が備えるハッシュ関数回路の実施例を示す図である。

【図 6】

本発明の第 2 の実施の形態によるデータ依存関係検出装置を含むマルチプロセッサの構成を示すブロック図である。

【図 7】

本発明の第 2 の実施の形態によるデータ依存関係検出装置の動作の具体例を説明する説明図である。

【図 8】

本発明の第 3 の実施の形態によるデータ依存関係検出装置の構成を示すブロック図である。

【図 9】

データ依存投機実行の動作を説明する説明図である。

【図 1 0】

従来のデータ依存関係検出装置の構成を示すブロック図である。

【符号の説明】

- 1 0 データ依存関係検出装置
- 1 1 ハッシュ関数回路
- 1 2 命令履歴テーブル
- 1 3 投機実行フラグ

- 1 4 実行命令
- 1 5 ロード・ストアアドレス
- 1 6 データ依存検出結果
- 1 7 プロセッサ制御部
- 1 8 命令実行部
- 2 0 データ依存関係検出装置
- 2 1 ハッシュ関数回路
- 2 2 命令履歴テーブル
- 3 0 データ依存関係検出装置
- 3 1 ハッシュ関数回路
- 3 2 命令履歴テーブル
- 3 3 投機実行フラグ
- 3 4 実行命令
- 3 5 ロード・ストアアドレス
- 3 6 データ依存検出結果
- 3 7 ロード・ストアアドレス
- 3 8 実行命令
- 3 9 投機実行フラグ
- 4 0 スレッド順序
- 4 1 論理和回路
- 4 5 ロード・ストアアドレス
- 4 6 実行命令
- 5 0、5 1、5 2、5 3 プロセッサ
- 5 5 スレッド制御部
- 5 7 プロセッサ制御部
- 6 0 データ依存関係検出装置
- 6 1 論理和回路
- 6 2 命令履歴テーブル
- 6 3 プロセッサ制御部

6 4 データ依存検出結果

1 0 0 データ依存関係検出装置

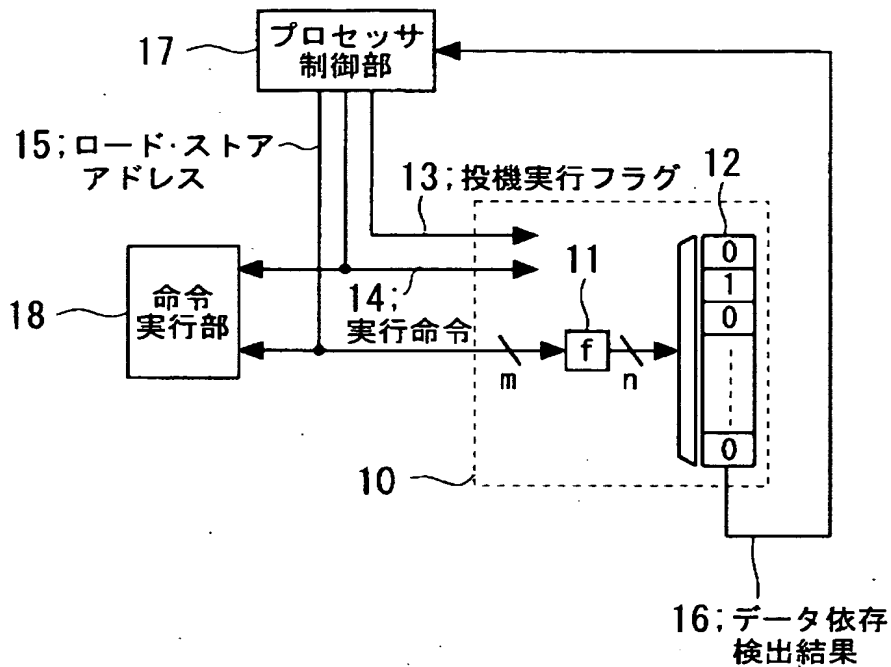
1 0 1 アドレスバッファ

1 0 2 アドレス比較器

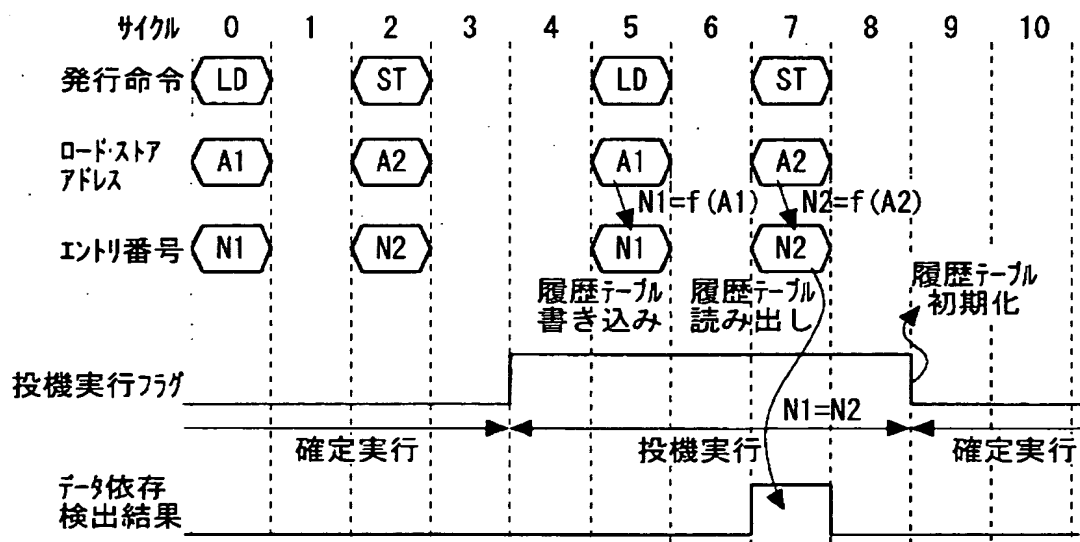
1 0 3 論理和回路

【書類名】 図面

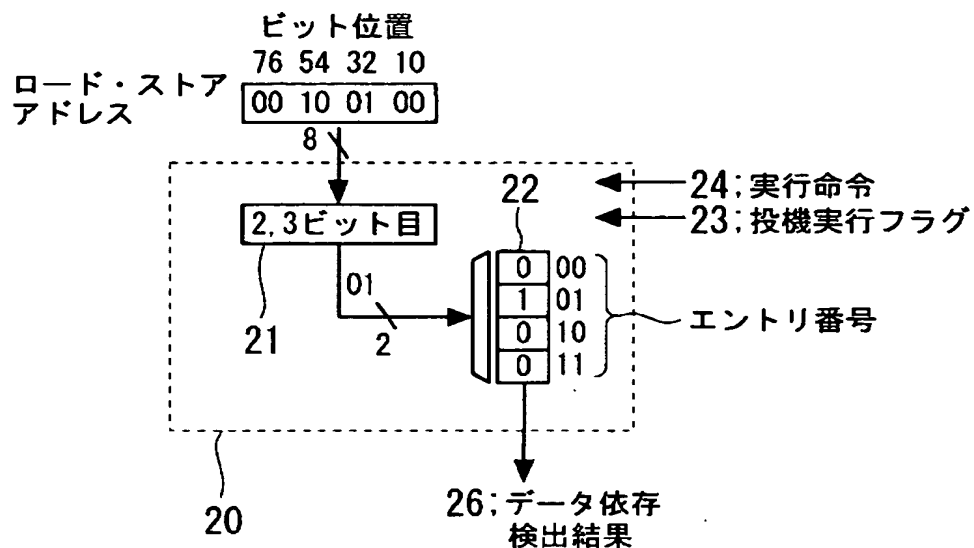
【図 1】



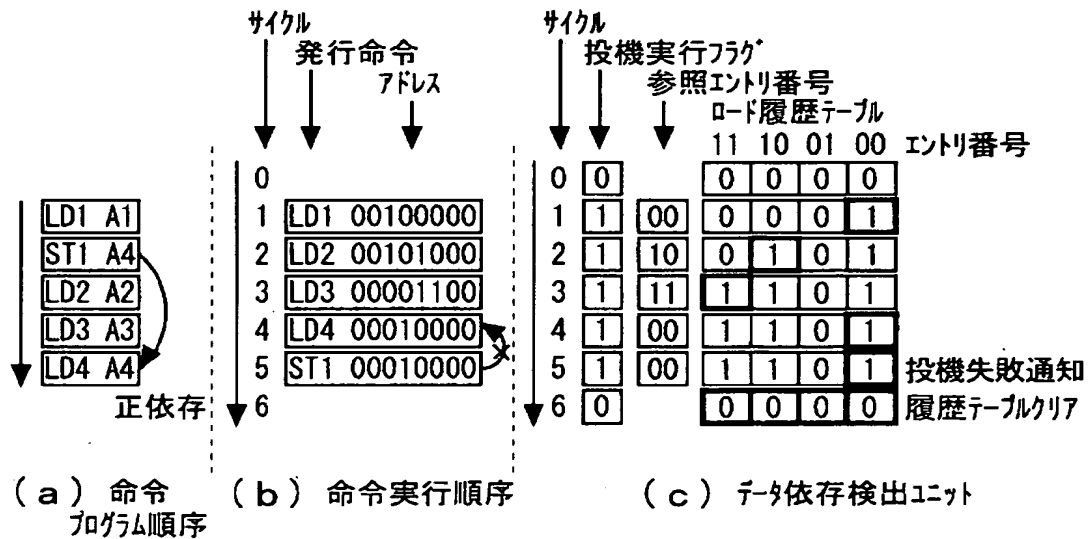
【図 2】



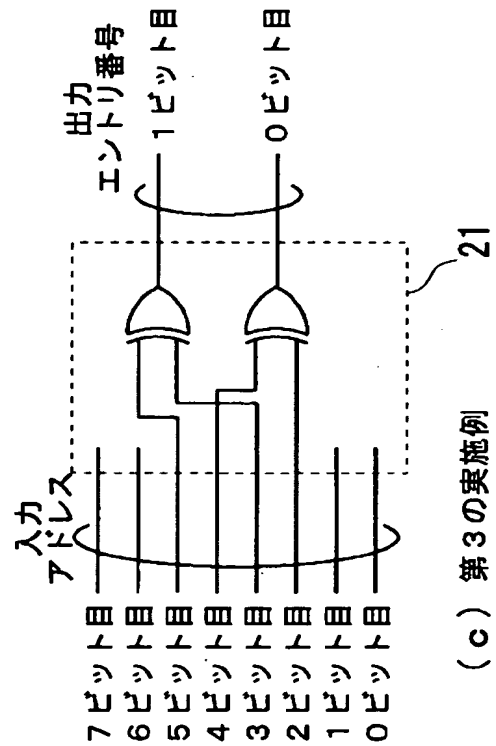
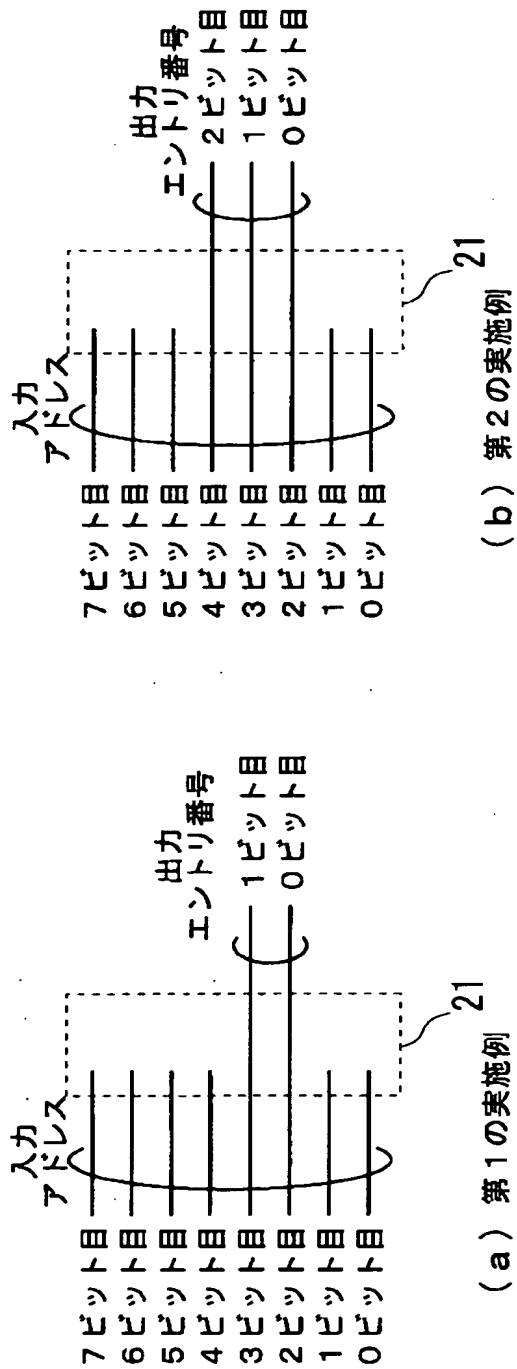
【図 3】



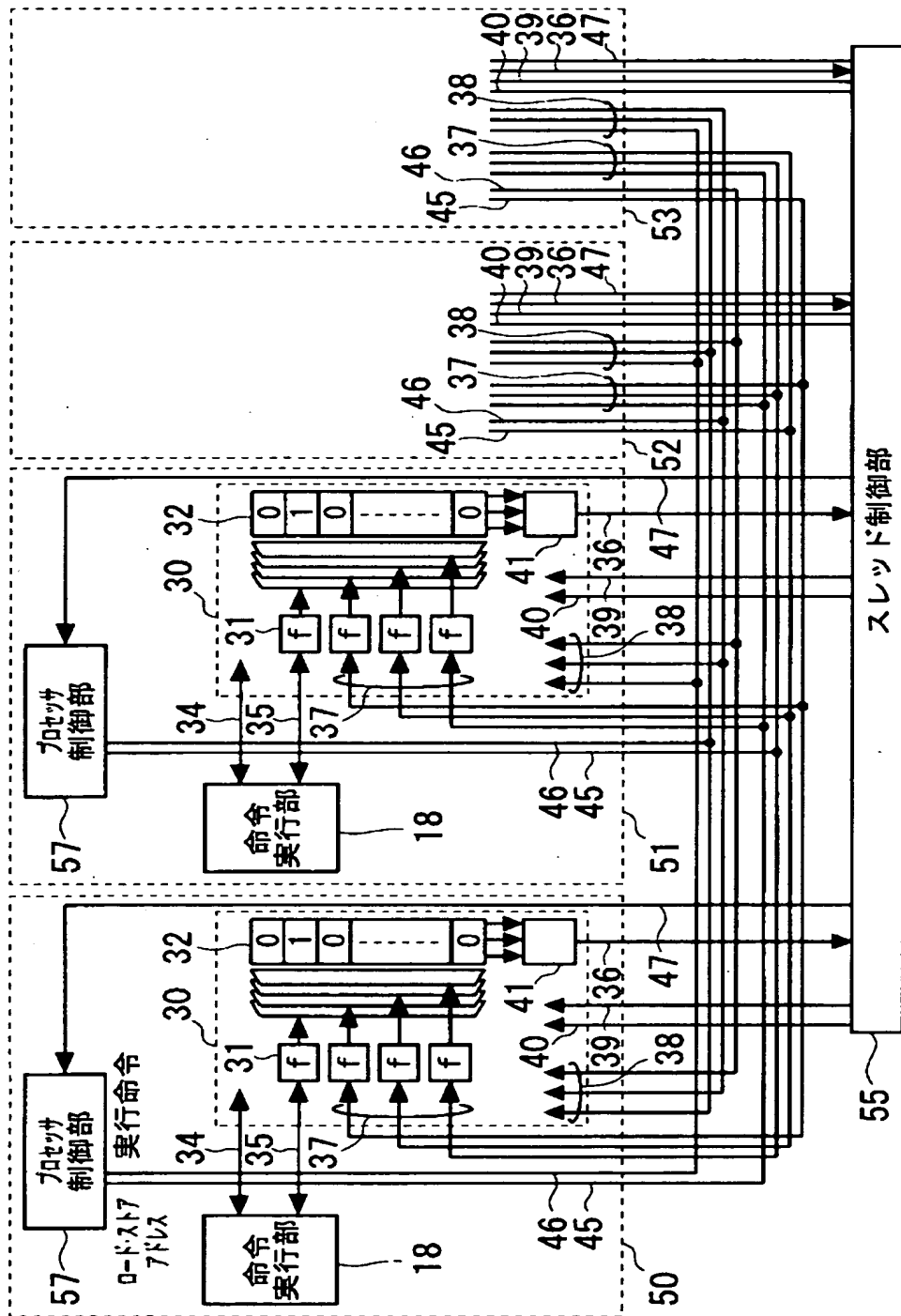
【図 4】



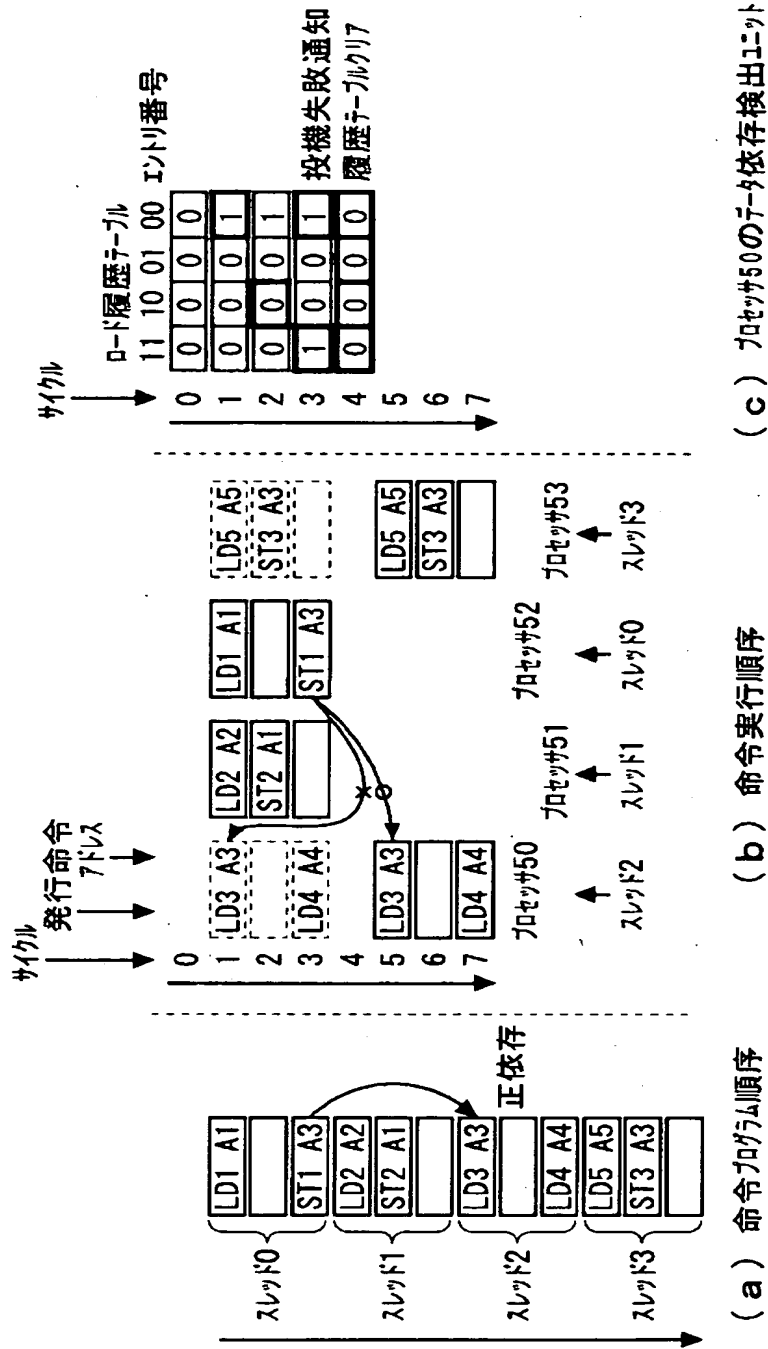
【図 5】



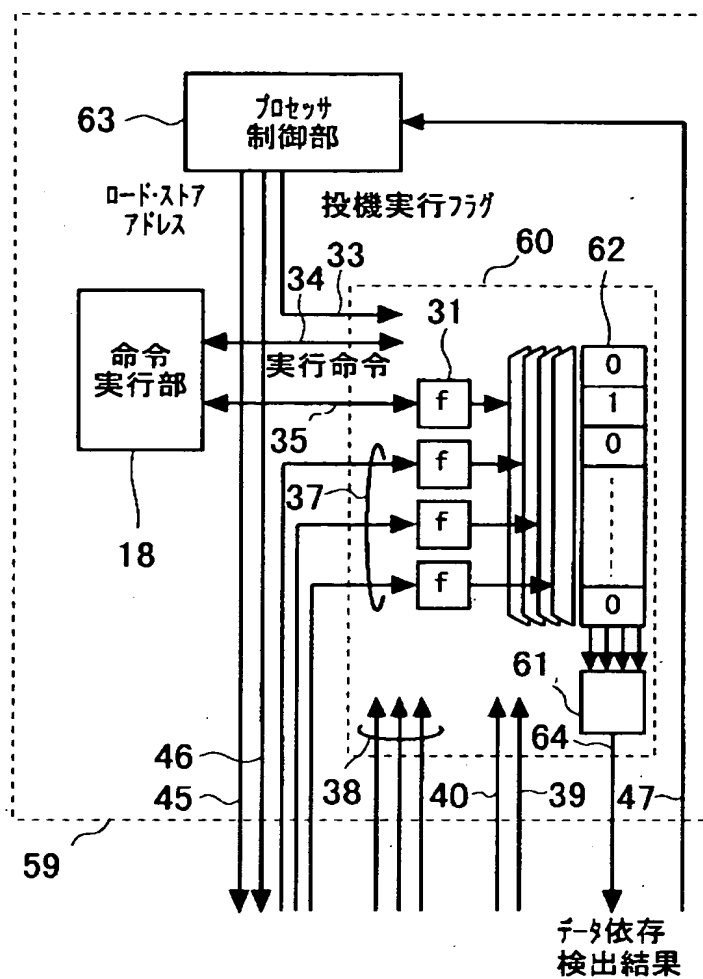
【図 6】



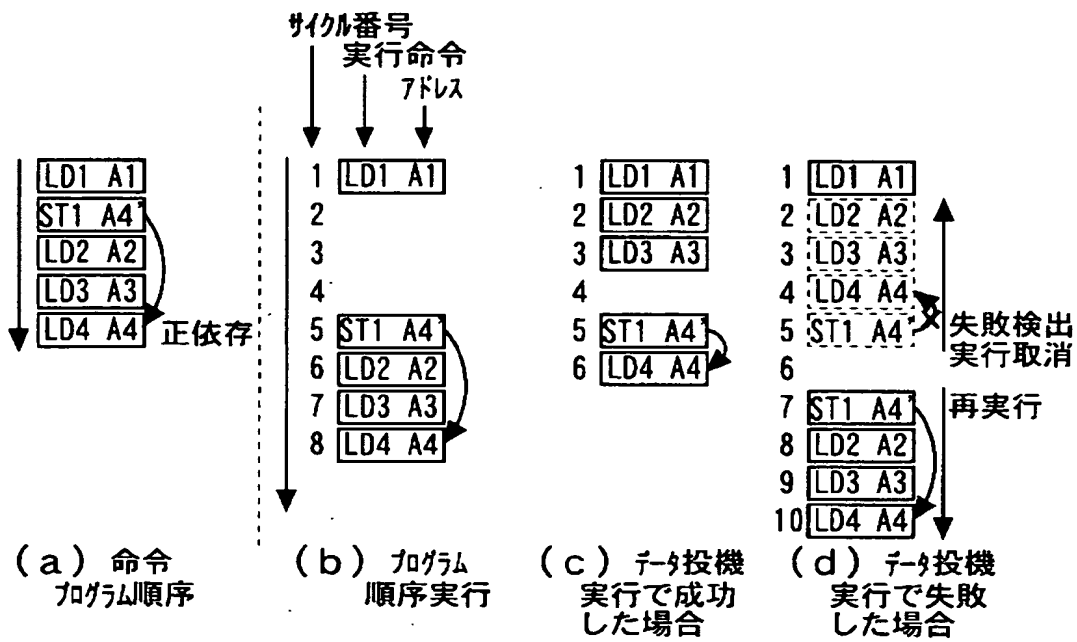
【図 7】



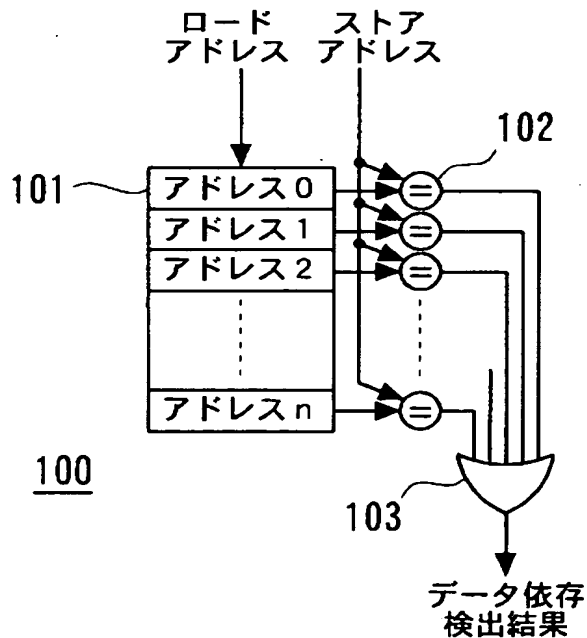
【図8】



【図 9】



【図 1 0】



【書類名】 要約書

【要約】

【課題】

データ依存投機実行することのできるロード命令の数が、ハードウェア量に制限されず、小さいハードウェア量でもデータ依存投機実行による実行性能の向上を享受することができ、依存検出の処理時間が短く、プロセッサの動作周波数を高速化することが容易である、データ依存関係検出装置の提供。

【解決手段】

プロセッサが非プログラム順序でロード命令またはストア命令の実行を行う際に、前記ロード命令またはストア命令が処理対象とするアドレス間の依存関係の存在の可能性を検出するデータ依存関係検出装置であって、実際に前記依存関係が存在する場合は必ず依存関係が存在する旨を検出するが、実際に前記依存関係が存在しない場合でも依存関係が存在する旨を検出する場合があることを許容することを特徴とし、ロード命令及びストア命令が処理対象とするアドレスを一意的な番号に変換するアドレス変換手段と、前記変換された番号に対応して、ロード命令又はストア命令が実行されたか否かを記憶する複数の記憶手段から構成される実行履歴記憶手段とを備える。

【選択図】

図 1

出 願 人 履 歴 情 報

識別番号 [000004237]

1. 変更年月日	1990年 8月29日
[変更理由]	新規登録
住 所	東京都港区芝五丁目7番1号
氏 名	日本電気株式会社